Harita Shroff
CS-156
Fall 2017
11/03/2017

Homework-3

1. Consider the problem of getting a BSCS at SJSU. Simplify the prerequisite graph of CS courses by only consider deep course sequences (i.e., CS157A, CS157B; ; etc) which have CS146 only as a prerequisite for the first course. Treat all these sequences as the same sequence CSDeepA and CSDeepB. Write down the the task of just scheduling your CS courses as a CSP. You should have sufficient constraints so that a solution would mean that you have satisfied all undergrad CS requirements (you can ignore general ed and other requirements). Your domains should be the possible semesters to start a course or DNT if the course was not taken. For example, 1 means you start the course the first semester, 2 the second, etc. Let 8 be the highest semester number and assume you can take at most 5 courses a semester.

- Variables = {CS49J, CS49C, CS46A, CS72, CS 47, CS46B, CS172A, CS147, CS174, CS144, CS154, CS130, CS146, CS151, CS175, CS100W, CS172B, CS166, CS149, CS122, CS155, CS159, CS134, CS156, CS152, CS160, CS143M, CS143C, CS 173, CSDeepA, CSDeepB}
- Assumptions:
    - Not considering Non “CS” boxes in the graph
    - Taking CSDeepA and CSDeepB to replace all deep courses
    - Not considering CS167A
    - “<” constraint indicates that value of the variable on the left hand side is less than the value on the right hand side.
    - Also assuming a student can only take 2 electives in given semester.

- Domain = {1, 2, 3, 4, 5, 6, 7, 8, DNT}
- Constraints = {
  # Constraints to maintain order between courses
  CSDeepA < CSDeepB, CS147 < CSDeepA, CS46A < CS46B, CS46A < CS72, CS46B < CS47, CS46B < CS174, CS46B < CSDeepA, CS46B < CS144, CS46B < CS154, CS46B < CS130, CS46B < CS146, CS46B < CS151, CS49J < CS46B, CS49J < CS175, CS72 < CS 172A, CS172A < CS 172B, CS172B < CS173, CS47 < CS147, CS47 < CS166, CS46A < CS143M, CS46A < CS143C, CS146 < CS166, CS146 < CS149, CS146 < CSDeepA, CS146 < CS122, CS146 < CS155, CS146 < CS159, CS146 < CS134, CS146 < CS156, CS146 < CS160, CS151 < CS134, CS151 < CS156, CS151 < CS152, CS151 < CS160, CS100W < CS 160, CS160 < CS161, CS49C < CS144

  # Constraints to confirm compulsory courses are not allowed value DNT

CS46A ≠ DNT, CS46B ≠ DNT, CS47 ≠ DNT, CS147 ≠ DNT, CS154 ≠ DNT, CS130 ≠ DNT, CS146 ≠ DNT, CS151 ≠ DNT, CS175 ≠ DNT, CS100W ≠ DNT, CS149 ≠ DNT, CS152 ≠ DNT, CS160 ≠ DNT, CSDeepA ≠ DNT, CS49C ≠ DNT

# Constraint for Electives
CS72 = (DNT OR BETWEEN(1, 4)),
CS172A = (DNT OR BETWEEN(2, 5)),
CS174 = (DNT OR BETWEEN(2,5)),
CS144 = (DNT OR BETWEEN(1,7),
CS172B = (DNT OR BETWEEN(3,6)),
CS166 = (DNT OR BETWEEN(3,8)),
CS122 = (DNT OR BETWEEN(3,8)),
CS155 = (DNT OR BETWEEN(3,8)),
CS159 = (DNT OR BETWEEN(3,8)),
CS134 = (DNT OR BETWEEN(3,8)),
CS156 = (DNT OR BETWEEN(3,8)),
CS143M = (DNT OR BETWEEN(1,4)),
CS143C = (DNT OR BETWEEN(1,4)),
CS173 = (DNT OR BETWEEN(5,8))

Constraint to select limited number of courses each semester
BETWEEN(1, CS46B, 4),
BETWEEN(1, CS175, 4),
BETWEEN(1, CS47, 5),
BETWEEN(1, CS146, 6)
BETWEEN(3, CS147, 6),
BETWEEN(4, CS149, 7),
BETWEEN(1, CS151, 7),
BETWEEN(1, CS130, 7),
BETWEEN(1, CS154, 7),
BETWEEN(5, CS152, 8),
BETWEEN(5, CS160, 8),
BETWEEN(5, CSDeepA, 8),
}

2. Let CS46B be the variable corresponding to the class CS46B. If we use the degree heuristic to choose the first variable in backtracking search to solve our CSP and if we break ties by picking the smaller numbered course, what is the variable X that would be considered in backtracking search in your CSP and what course does it correspond to? What would be the result of the REVISE(csp, X,CS46B)?
    - X = CS46A it corresponds to CS46A
      Domains before calling REVISE,

- Domain of CS46A = {1, 2, 3, 4, 5, 6, 7, 8}
  Domain of CS46B = {2, 3}

Domains After calling REVISE,

- Domain of CS46A = {1} as Constraint suggests (CS46A < CS46B)
- Domain of CS46B = {2, 3}

3. Suppose we choose CS49C to be taken in the first semester. Show all of the domains for each course after doing an arc consistency check (you only need to list the ones that change), in doing the check carefully trace the AC3 algorithm. In terms of works you did, how did this compare to the REVISE operation of the previous problem.
   - CS49C = {1}
   - CS144 = {DNT, 2, 3, 4, 5, 6, 7}
   - CS46B = {2}
   - CS146 = {3,4,5}

4. From this updated CSP, using the MINIMUM REMAINING VALUE heuristic, give the variable to consider next. Break ties by choosing the course of smaller number.
   - CS146 is the next variable to consider as,
     - It has 3 possible legal values
     - {3, 4, 5}

5. Using the variable from the last step, determine the value for it using the LEAST-CONSTRAINING-VARIABLE heuristic breaking ties by picking the earliest possible semester.
   - Selecting CS146={5},
     - CS166 = {DNT, 6, 7}
     - CS149 = {6}
     - CS122 = {DNT, 6, 7}
     - CS155 = {DNT, 6, 7}
     - CS159 = {DNT, 6, 7}
     - CS134 = {DNT, 6, 7}
     - CS156 = {DNT, 6, 7}

6. This next problem is unconnected with the five previous. Come up with a logical formula which expresses P3,3, there is a pit in square (3,3), in terms of the variables (more than one) Bx,y expressing there is a breeze in square (x,y). Convert this formula to CNF by hand by first doing biconditional elimination, then implication elimination, then using distributivity. Check your work by downloading the software from Russell and Norvig (the book's code requires Python3), launching the Python interpreter, typing from logic import * then doing to_cnf("your original formula"). Obviously, replace "your original formula" with the formula you had before converting to CNF. Have the transcript of your interaction with Python as part of your solution.

- Logical formula: P33 ⇔ (B23 ^ B32 ^ B34 ^ B43)

  By Biconditional Elimination,
  P33 => (B23 ^ B32 ^ B34 ^ B43) ^ (B23 ^ B32 ^ B34 ^ B43) => P33

  By Implication Elimination,
  (~P33 V (B23 ^ B32 ^ B34 ^ B43)) ^ (~(B23 ^ B32 ^ B34 ^ B43) V P33)

  By DeMorgan's rule
  (~P33 V (B23 ^ B32 ^ B34 ^ B43)) ^ ((~B23 V ~B32 V ~B34 V ~B43) V P33)

  By Distributivity,
  (~P33 V B23) ^ (~P33 V B32) ^ (~P33 V B34) ^ (~P33 V B43) ^
  (~B23 V ~B32 V ~B34 V ~B43 V P33)

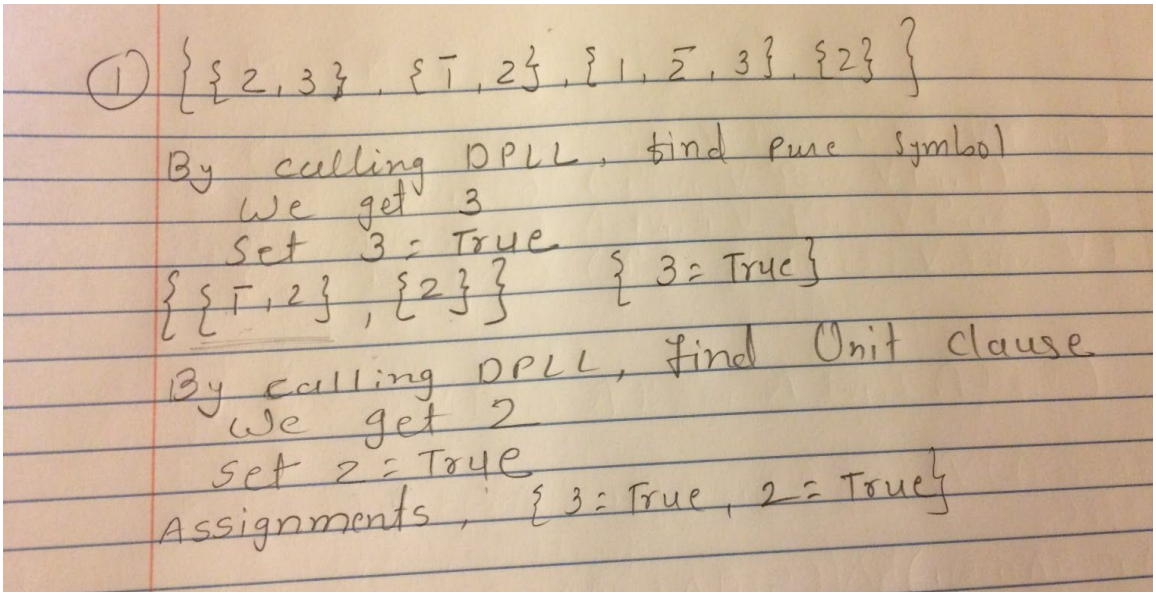
- Transcript

```
hshroff:aima-python-master hshroff$ python3
Python 3.6.3 (default, Oct  3 2017, 06:09:15)
[GCC 4.2.1 Compatible Apple LLVM 9.0.0 (clang-900.0.37)] on darwin
Type "help", "copyright", "credits" or "license" for more information.
>>> from logic import *
>>> to_cnf("P33 <=> B23 & B32 & B34 & B43")
((P33 | ~B23 | ~B32 | ~B34 | ~B43) & (B23 | ~P33) & (B32 | ~P33) & (B34 | ~P33) & (B43 | ~P33))
```

2.

A.

```
hshroff:Hw3 hshroff$ python cnf_generator.py 3 3 3
c
c Random CNF in satcompetition.org format created by cnf_generator.php
c
p cnf 3 3
2 3 0
-1 2 0
1 -2 3 0
2 0
```

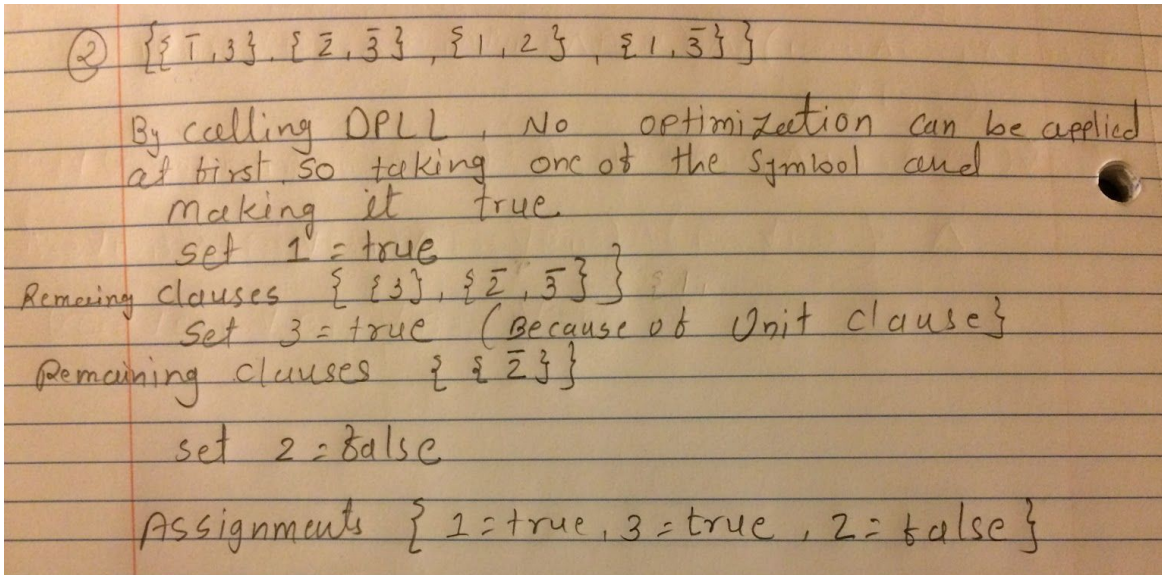



```
hshroff:Hw3 hshroff$ python sat_solver.py testsample.txt
Clauses
[[2, 3], [-1, 2], [1, -2, 3], [2]]
Symbols
{1: 1, 2: 2, 3: 3, -2: -2, -1: -1}
Assignment
{}
Symbols
{1: 1, 2: 2, -1: -1, -2: -2}
Assignment
{3: 1, -3: -1}
Symbols
{1: 1, -1: -1}
Assignment
{2: 1, 3: 1, -3: -1, -2: -1}
{2: 1, 3: 1, -3: -1, -2: -1}
```

B.

```
hshroff:Hw3 hshroff$ python cnf_generator.py 3 3 3
c
c Random CNF in satcompetition.org format created by cnf_generator.php
c
p cnf 3 3
-1 3 0
-2 -3 0
```

1 2 0
1 -3 0

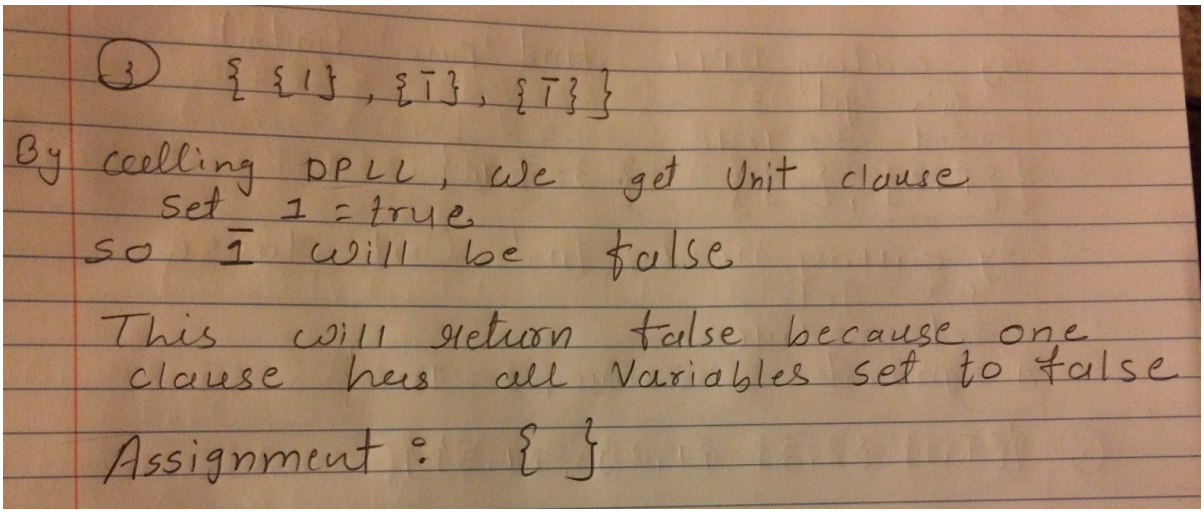


hshroff:Hw3 hshroff$ python sat_solver.py testsample.txt
Clauses
[[-1, 3], [-2, -3], [1, 2], [1, -3]]
Symbols
{1: 1, 2: 2, 3: 3, -1: -1, -3: -3, -2: -2}
Assignment
{}
Symbols
{2: 2, 3: 3, -3: -3, -2: -2}
Assignment
{1: 1, -1: -1}
Symbols
{2: 2, -2: -2}
Assignment
{1: 1, 3: 1, -3: -1, -1: -1}
Symbols
{}
Assignment
{1: 1, 2: -1, 3: 1, -2: 1, -3: -1, -1: -1}
{1: 1, 2: -1, 3: 1, -2: 1, -3: -1, -1: -1}

C.

hshroff:Hw3 hshroff$ python cnf_generator.py 1 2 2
c

c Random CNF in satcompetition.org format created by cnf_generator.php
c
p cnf 1 2
1 0
-1 0
-1 0



hshroff:Hw3 hshroff$ python sat_solver.py testsample.txt
Clauses
[[1], [-1], [-1]]
Symbols
{1: 1, -1: -1}
Assignment
{}
Symbols
{}
Assignment
{1: 1, -1: -1}
False