cpts350 Symbolic graph project

0. Make yourself be familiar with Python and `pyEDA` package (see the email that I sent earlier this week and read the example code in the documentation of the package). You may find installation instructions at

`https://pyeda.readthedocs.io/en/latest/install.html`

1. Look at your class notes on how a graph is represented in a Boolean formula and then a Boolean formula is represented in BDD, and on how the transitive closure is computed, in particular, looking at the example of computing the transitive closure of $R \circ R$.

2. Let G be a graph over 32 nodes (namely, node 0, $\cdots$, node 31). For all $0 \leq i, j \leq 31$, there is an edge from node i to node j iff $(i+3)\%32 = j\%32$ or $(i+8)\%32 = j\%32$. (% is the modular operator in C; e.g., 35% 32=3.) A node i is `even` if i is an even number. A node i is `prime` if i is a prime number. In particular, we define [`even`] as the set $\{0, 2, 4, 6, \cdots, 30\}$ and [`prime`] as the set $\{3, 5, 7, 11, 13, 17, 19, 23, 29, 31\}$. We use R to denote the set of all edges in G.

3. (graded on correctness and clarity. If you use explicit graph search such as DFS, you receive 0.) (coding in Python) Every finite set can be coded as a BDD. Please write a Python program to decide whether the following is true:

(StatementA) for each node u in [`prime`], there is a node v in [`even`] such that u can reach v in a positive even number of steps.

Your code shall implement the following steps.

step3.1. Obtain BDDs RR, $EVEN$, $PRIME$ for the finite sets R, [`even`], [`prime`] , respectively. Pay attention to the use of BDD variables in your BDDs. Your code shall also verify the following test cases:

$RR(27, 3)$ is true;
$RR(16, 20)$ is false;
$EVEN(14)$ is true;
$EVEN(13)$ is false;
$PRIME(7)$ is true;
$PRIME(2)$ is false.

step3.2. Compute BDD $RR2$ for the set $R \circ R$, from BDD RR. Herein, $RR2$ encodes the set of node pairs such that one can reach the other in two steps. Your code shall also verify the following test cases:

$RR2(27, 6)$ is true;
$RR2(27, 9)$ is false.

step3.3. Compute the transitive closure $RR2star$ of $RR2$. Herein, $RR2star$ encodes the set of all node pairs such that one can reach the other in a positive even number of steps.

step3.4. Here comes the most difficult part. We first StatementA formally:

$$\forall u.\ (PRIME(u) \rightarrow \exists v.\ (EVEN(v) \wedge RR2star(u, v))).$$

There are two quantifiers in StatementA: one is "for each", and the other is "there is". First, from what you have learned from math216 (discrete math), "for each" can be expressed through "there is". Second, "there is" can be implemented using existential quantifier elimination method $BDD.smoothing()$. As a result, the entire StatementA is a BDD without free variables and hence it is either true or false. Return the truth value.

Many students find methods $BDD.compose()$ and $BDD.smoothing()$ are quite useful in the package.

4. You need turn-in working code, screen-shot of code execution results. Make sure that you put comments along with your code so it is readable. Your TAs will probably run your code!

Remark: The mathematics behind this project is from Frobenius number in Number Theory. You may look at the wiki page of the Coin Problem:

`https://en.wikipedia.org/wiki/Coin_problem`

if you like number theory. The name shall sound familiar to you: in a homework problem that you just did, you used the logorithm of the Perron-Frobenius number of a matrix (the largest eigenvalue, which happens to be a nonnegative real, of a nonnegative matrix) to estimate the asymptoic growth rate of the number of walks with length n in a graph represented as the matrix.