University of Waterloo

ECE 653 – Software Testing, Quality Assurance and Maintenance

Winter 2018

Assignment 2 Report

Maria N. Samad – 20669032 (mnsamad@uwaterloo.ca)

Due Date: March 12th, 2018

QUESTION # 1:

(a) There are four paths that Prog1 can have, which are as follows:

Path 1: 1, 2, 3, 4, 8, 9, 10, 11, 15
Path 2: 1, 2, 3, 4, 8, 9, 12, 13, 14, 15
Path 3: 1, 2, 5, 6, 7, 8, 9, 10, 11, 15
Path 4: 1, 2, 5, 6, 7, 8, 9, 12, 13, 14, 15

(b) **Path 1:** 1, 2, 3, 4, 8, 9, 10, 11, 15

Edge	Symbolic State (PV)	Path Condition (PC)
1 -> 2	$x \mapsto X_0 , y \mapsto Y_0$	true
2 -> 3	$x \mapsto X_0 , y \mapsto Y_0$	$X_0 - Y_0 > 30$
3 -> 4	$x \mapsto X_0 + 1, y \mapsto Y_0$	$X_0 - Y_0 > 30$
4 -> 8	$x \mapsto X_0 + 1, y \mapsto Y_0 - 2$	$X_0 - Y_0 > 30$
8 -> 9	$x \mapsto X_0 + 1, y \mapsto Y_0 - 2$	$X_0 - Y_0 > 30$
9 -> 10	$x \mapsto X_0 + 1, y \mapsto Y_0 - 2$	$X_0 - Y_0 > 30 \land 3 * (X_0 - Y_0 + 3) < 40$
10 -> 11	$x \mapsto (X_0 + 1) * 3, y \mapsto Y_0 - 2$	$X_0 - Y_0 > 30 \land 3 * (X_0 - Y_0 + 3) < 40$
11 -> 15	$x \mapsto (X_0 + 1) * 3, y \mapsto (Y_0 - 2) * 2$	$X_0 - Y_0 > 30 \land 3 * (X_0 - Y_0 + 3) < 40$

Path 2: 1, 2, 3, 4, 8, 9, 12, 13, 14, 15

Edge	Symbolic State (PV)	Path Condition (PC)
1 -> 2	$x \mapsto X_0 , y \mapsto Y_0$	true
2 -> 3	$x \mapsto X_0 , y \mapsto Y_0$	$X_0 - Y_0 > 30$
3 -> 4	$x \mapsto X_0 + 1, y \mapsto Y_0$	$X_0 - Y_0 > 30$
4 -> 8	$x \mapsto X_0 + 1, y \mapsto Y_0 - 2$	$X_0 - Y_0 > 30$
8 -> 9	$x \mapsto X_0 + 1, y \mapsto Y_0 - 2$	$X_0 - Y_0 > 30$
9 -> 12	$x \mapsto X_0 + 1, y \mapsto Y_0 - 2$	$X_0 - Y_0 > 30 \land 3 * (X_0 - Y_0 + 3) >= 40$
12 -> 13	$x \mapsto X_0 + 1, y \mapsto Y_0 - 2$	$X_0 - Y_0 > 30 \land 3 * (X_0 - Y_0 + 3) >= 40$
13 -> 14	$x \mapsto (X_0 + 1) * 4, y \mapsto Y_0 - 2$	$X_0 - Y_0 > 30 \land 3 * (X_0 - Y_0 + 3) >= 40$
14 -> 15	$x \mapsto (X_0 + 1) * 4, y \mapsto (Y_0 - 2) * 7$	$X_0 - Y_0 > 30 \land 3 * (X_0 - Y_0 + 3) >= 40$

Path 3: 1, 2, 5, 6, 7, 8, 9, 10, 11, 15

Edge	Symbolic State (PV)	Path Condition (PC)
1 -> 2	$x \mapsto X_0 , y \mapsto Y_0$	true
2 -> 5	$x \mapsto X_0 , y \mapsto Y_0$	$X_0 - Y_0 <= 30$
5 -> 6	$x \mapsto X_0 , y \mapsto Y_0$	$X_0 - Y_0 <= 30$
6 -> 7	$x \mapsto X_0 , y \mapsto Y_0 + 5$	$X_0 - Y_0 <= 30$
7 -> 8	$x \mapsto X_0 - 3, y \mapsto Y_0 + 5$	$X_0 - Y_0 <= 30$
8 -> 9	$x \mapsto X_0 - 3, y \mapsto Y_0 + 5$	$X_0 - Y_0 <= 30$
9 -> 10	$x \mapsto X_0 - 3, y \mapsto Y_0 + 5$	$X_0 - Y_0 <= 30 \land 3 * (X_0 - Y_0 - 8) < 40$
10 -> 11	$x \mapsto (X_0 - 3) * 3, y \mapsto Y_0 + 5$	$X_0 - Y_0 <= 30 \land 3 * (X_0 - Y_0 - 8) < 40$
11 -> 15	$x \mapsto (X_0 - 3) * 3, y \mapsto (Y_0 + 5) * 2$	$X_0 - Y_0 <= 30 \land 3 * (X_0 - Y_0 - 8) < 40$

Path 4: 1, 2, 5, 6, 7, 8, 9, 12, 13, 14, 15

Edge	Symbolic State (PV)	Path Condition (PC)
1 -> 2	$x \vdash> X_0$, $y \vdash> Y_0$	true
2 -> 5	$x \vdash> X_0$, $y \vdash> Y_0$	$X_0 - Y_0 <= 30$
5 -> 6	$x \vdash> X_0$, $y \vdash> Y_0$	$X_0 - Y_0 <= 30$
6 -> 7	$x \vdash> X_0$, $y \vdash> Y_0 + 5$	$X_0 - Y_0 <= 30$
7 -> 8	$x \vdash> X_0 - 3$, $y \vdash> Y_0 + 5$	$X_0 - Y_0 <= 30$
8 -> 9	$x \vdash> X_0 - 3$, $y \vdash> Y_0 + 5$	$X_0 - Y_0 <= 30$
9 -> 12	$x \vdash> X_0 - 3$, $y \vdash> Y_0 + 5$	$X_0 - Y_0 <= 30 \land 3 * (X_0 - Y_0 - 8) >= 40$
12 -> 13	$x \vdash> X_0 - 3$, $y \vdash> Y_0 + 5$	$X_0 - Y_0 <= 30 \land 3 * (X_0 - Y_0 - 8) >= 40$
13 -> 14	$x \vdash> (X_0 - 3) * 4$, $y \vdash> Y_0 + 5$	$X_0 - Y_0 <= 30 \land 3 * (X_0 - Y_0 - 8) >= 40$
14 -> 15	$x \vdash> (X_0 - 3) * 4$, $y \vdash> (Y_0 + 5) * 7$	$X_0 - Y_0 <= 30 \land 3 * (X_0 - Y_0 - 8) >= 40$

(c)

Path 1: 1, 2, 3, 4, 8, 9, 10, 11, 15

Not feasible, because the path conditions require the difference to be greater than 30, and then the product term to be less than 40, so not possible to have specific values for x and y like that

Path 2: 1, 2, 3, 4, 8, 9, 12, 13, 14, 15

A feasible path, for example $X_0 = 31$ and $Y_0 = 0$. It satisfies the two conditions in the path.

Path 3: 1, 2, 5, 6, 7, 8, 9, 10, 11, 15

This is also a feasible path, which will satisfy all the path conditions. For example $X_0 = 0$, and $Y_0 = 0$

Path 4: 1, 2, 5, 6, 7, 8, 9, 12, 13, 14, 15

Another feasible path for $X_0 = 30$ and $Y_0 = 0$

QUESTION # 2:

(a) The constraint at-most-one(a1, . . . , an) is satisfied if at most one of the Boolean variables a1, . . . , an is true.

The constraint at-most-one(a1, a2, a3, a4) can be defined into an equivalent set of clauses, as follows:

- a1 **V** ˥ a2 **V** ˥ a3 **V** ˥ a4
- ˥ a1 **V** a2 **V** ˥ a3 **V** ˥ a4
- ˥ a1 **V** ˥ a2 **V** a3 **V** ˥ a4
- ˥ a1 **V** ˥ a2 **V** ˥ a3 **V** a4

(b) v_{init} **V** $(\bigvee_{i=1}^{n}(v_{i,i+1}))$ **V** v_{end}

(c) Extending code for the real data:

$$\bigvee_{0<i<n} (\boldsymbol{a_{n-i}} \ \mathbf{V} \ \boldsymbol{a_i})$$

$$\bigvee_{i=1}^{n}\left(\boldsymbol{a_i}\ \mathbf{V}\ \boldsymbol{a_i}\left(\bigvee_{j=1}^{n}(\neg\ \boldsymbol{a_j})\right)\right)$$

QUESTION # 3:

(a) D + E = Y

N + R + c1 = E

E + O + c2 = N

S + M + c3 = O

c4 = M

Here, we know M cannot be 0 otherwise, it wouldn't have carried over. It can only be 1 because the highest two single digit numbers (9) even when added will result in a value with carry as 1.
Similarly, S and M cannot be 0 as they are the leading digits of the two numbers. Z3 solver is then used to deduce the values for the constraints

(b) The code in verbal_arithmetic.py has been updated to define the constraints as deduced by using the Z3 Solver functionality. Refer to the updated file a2`q3/verbal_arithmetic.py` on GitHub for the detailed solution.

(c) Refer to the updated file a2`q3/puzzle_tests.py` on GitHub for the complete solution.

QUESTION # 4:

The code is giving me some problems in running, but please check the coding at least for marks.

Check the sym.py, tet_sym, run.py for the code

Coverage is index_Q4d.html

QUESTION # 5:

(a) The statement is valid. Let's say we have a model $M = (S, P^M, Q^M)$.
Assume $S=\{1,2\}$, such that $P(1) = T$ and $P(2) = F$, and $Q(1) = F$ and $Q(2) = T$, i.e. $P = \{1\}$ and $Q = \{2\}$

Let's solve by contradiction. If $M1 \models \forall x . \exists y . P(x) \lor Q(y)$, then by contradiction $M2 \not\models \forall x.P(x) \lor \exists y.Q(y)$, and vice versa. So if M1 is not valid, M2 should not be valid. Apply negation to LHS -> RHS

$\neg ((\forall x . \exists y . P(x) \lor Q(y)) \Rightarrow (\forall x.P(x) \lor \exists y.Q(y)))$
By negation and implication rules, we get:
$\neg (\forall x . \exists y . P(x) \lor Q(y)) \lor (\forall x.P(x) \lor \exists y.Q(y))$
Applying DeMorgan's and Quantifier Negations on the first part to get:
$\exists x . \forall y . \neg P(x) \land \exists x . \forall y . \neg Q(y) \lor \forall x . P(x) \lor \exists y . Q(y)$
The first two terms can be reduced as:
$\exists x . \neg P(x) \land \forall y . \neg Q(y) \lor \forall x . P(x) \lor \exists y . Q(y)$

For $S=\{1,2\}$, $P(1) = T$ and $Q(2) = T$. Applying this to the sentence we get:
$\exists x . \neg P(x) \land \forall y . \neg Q(y) \lor \forall x . P(x) \lor \exists y . Q(y) = T \land T \lor F \lor T = T$
This is because, there's a value (2) in the set for which P is not true, so the term becomes valid. Similarly, there's a value (1) in the set for which Q is not true, so even this term becomes valid. However, P is not true for all x's so that's False, and lastly, there's a value (2) in the set for which Q is true, so even that is valid, and you get True at the end.

Now looking at the other way round, i.e. RHS -> LHS, and apply negation:

$\neg ((\forall x.P(x) \lor \exists y.Q(y)) \Rightarrow (\forall x . \exists y . P(x) \lor Q(y))$
By negation and implication rules, we get:
$\neg (\forall x.P(x) \lor \exists y.Q(y)) \lor (\forall x . \exists y . P(x) \lor Q(y))$
Applying DeMorgan's and Quantifier Negations on the first part to get:
$\exists x . \neg P(x) \land \forall y . \neg Q(y) \lor \forall x . (P(x) \lor \exists y . Q(y))$
The last two terms can be reduced as:
$\exists x . \neg P(x) \land \forall y . \neg Q(y) \lor \forall x . P(x) \lor \exists y . Q(y)$

For $S=\{1,2\}$, $P(1) = T$ and $Q(2) = T$. Applying this to the sentence we get:
$\exists x . \neg P(x) \land \forall y . \neg Q(y) \lor \forall x . P(x) \lor \exists y . Q(y) = T \land T \lor F \lor T = T$
This is because, there's a value (2) in the set for which P is not true, so the term becomes valid. Similarly, there's a value (1) in the set for which Q is not true, so even this term becomes valid. However, P is not true for all x's so that's False, and lastly, there's a value (2) in the set for which Q is true, so even that is valid, and you get True at the end.

This proves that the sentence is valid.

(b) The statement is not valid. Let's say we have a model $M = (S, P^M, Q^M)$.
Assume $S=\{(1,2),(3,4)\}$, such that $P(1,2) = T$ and $P(3,4) = F$, and $Q(1,2) = F$ and $Q(3,4) = T$, i.e. $P = \{(1,2)\}$ and $Q = \{(3,4)\}$

Let $M1 \models \forall x . \exists y . (P(x,y) \lor Q(x,y))$ and $M2 \models (\forall x . \exists y . P(x,y)) \lor (\forall x . \exists y . Q(x,y))$

Solve by contradiction, if valid M1 $\neq$ M2 should not hold true.

Apply negation to resolve implication:
$\neg (\forall x . \exists y . (P(x,y) \lor Q(x,y))) \lor (\forall x . \exists y . P(x,y)) \lor (\forall x . \exists y . Q(x,y))$
Applying DeMorgan's and Quantifier Negations on the first part to get:
$\exists x . \forall y . \neg P(x,y) \land \exists x . \forall y . \neg Q(x,y) \lor (\forall x . \exists y . P(x,y)) \lor (\forall x . \exists y . Q(x,y))$

For S={(1,2),(3,4)}, P(1,2) = T and Q(3,4) = T. Applying this to the sentence we get:
$\exists x . \forall y . \neg P(x,y) \land \exists x . \forall y . \neg Q(x,y) \lor (\forall x . \exists y . P(x,y)) \lor (\forall x . \exists y . Q(x,y)) = F \land F \lor F \lor F = F$
This is because, P(x,y) is not false for all y's, it is true for y=2, but not y=4, so the term becomes invalid. Similarly, Q(x,y) is not false for all y's, it is true for y=4, but not y=2, so the term becomes invalid. Also, P(x,y) is not true for all x's so that's False, and lastly, Q(x,y) is also not true for all x's, hence false, and you get False at the end.

(c) $\exists x . \exists y . (x \neq y) \land P(x) \land P(y) \land \forall s\, P(s) \Rightarrow (s - x) \lor (s - y)$
Where x, y are variables and s refers to the statement in the model

(d) $Array(A) \land (\forall i\, (0 \leq i < len(A)) \Rightarrow (\exists j . ((j < i) \land (A[j] < A[i]))) \land (\exists k . ((k > i) \land (A[k] > A[i]))))$

(e) $Array(A) \land \{\forall i . \forall j . (0 \leq i < j < len(A)) \Rightarrow A[i] \neq A[j]\}$