a2_sub

Student Name: Yichu Li
Student ID: 20817846
Email: y2792li@uwaterloo.ca

Q1

(a)

There are 4 execution paths in Prog1.
The first 'if' statement at Line 2 has 2 paths: x + y > 15 is true and x + y > 15 is false.
The second 'if' statement at Line 11 has 2 paths: 2 * (x + y) > 21 is true and 2 * (x + y) > 21 is false.
Therefore, the number of the total paths from Prog1 is 2 * 2 = 4.
Here are the 4 paths:
Path 1 (both "if" are true): 1,2,3,4,9,11,12,13,17.
Path 2 (first "if" is true, second is false): 1,2,3,4,9,11,14,15,16,17.
Path 3 (first "if" is false, second is true): 1,2,5,6,7,9,11,12,13,17.
Path 4 (both "if" are false): 1,2,5,6,7,9,11,14,15,16,17.

(b)

Path 1 (both "if" are true):

Edge	Symbolic State (PV)	Path Condition (PC)
$1 \rightarrow 2$	$x \mapsto X_0, y \mapsto Y_0$	true
$2 \rightarrow 3$	$x \mapsto X_0, y \mapsto Y_0$	$X_0 + Y_0 > 15$
$3 \rightarrow 4$	$x \mapsto X_0 + 7, y \mapsto Y_0$	$X_0 + Y_0 > 15$
$4 \rightarrow 9$	$x \mapsto X_0 + 7, y \mapsto Y_0 - 12$	$X_0 + Y_0 > 15$
$9 \rightarrow 11$	$x \mapsto X_0 + 9, y \mapsto Y_0 - 12$	$X_0 + Y_0 > 15$
$11 \rightarrow 12$	$x \mapsto X_0 + 9, y \mapsto Y_0 - 12$	$X_0 + Y_0 > 15 \wedge 2 * (X_0 + Y_0 - 3) > 21$
$12 \rightarrow 13$	$x \mapsto 3(X_0 + 9), y \mapsto Y_0 - 12$	$X_0 + Y_0 > 15 \wedge 2 * (X_0 + Y_0 - 3) > 21$
$13 \rightarrow 17$	$x \mapsto 3(X_0 + 9), y \mapsto 2(Y_0 - 12)$	$X_0 + Y_0 > 15 \wedge 2 * (X_0 + Y_0 - 3) > 21$

Path 2 (first "if" is true, second is false):

Edge	Symbolic State (PV)	Path Condition (PC)
$1 \rightarrow 2$	$x \mapsto X_0, y \mapsto Y_0$	true
$2 \rightarrow 3$	$x \mapsto X_0, y \mapsto Y_0$	$X_0 + Y_0 > 15$
$3 \rightarrow 4$	$x \mapsto X_0 + 7, y \mapsto Y_0$	$X_0 + Y_0 > 15$
$4 \rightarrow 9$	$x \mapsto X_0 + 7, y \mapsto Y_0 - 12$	$X_0 + Y_0 > 15$
$9 \rightarrow 11$	$x \mapsto X_0 + 9, y \mapsto Y_0 - 12$	$X_0 + Y_0 > 15$
$11 \rightarrow 14$	$x \mapsto X_0 + 9, y \mapsto Y_0 - 12$	$X_0 + Y_0 > 15 \wedge 2 * (X_0 + Y_0 - 3) \leq 21$
$14 \rightarrow 15$	$x \mapsto X_0 + 9, y \mapsto Y_0 - 12$	$X_0 + Y_0 > 15 \wedge 2 * (X_0 + Y_0 - 3) \leq 21$
$15 \rightarrow 16$	$x \mapsto 4(X_0 + 9), y \mapsto Y_0 - 12$	$X_0 + Y_0 > 15 \wedge 2 * (X_0 + Y_0 - 3) \leq 21$
$16 \rightarrow 17$	$x \mapsto 4(X_0 + 9), y \mapsto 3(Y_0 - 12) + 4(X_0 + 9)$	$X_0 + Y_0 > 15 \wedge 2 * (X_0 + Y_0 - 3) \leq 21$

Path 3 (first "if" is false, second is true):

Edge	Symbolic State (PV)	Path Condition (PC)
$1 \rightarrow 2$	$x \mapsto X_0, y \mapsto Y_0$	true
$2 \rightarrow 5$	$x \mapsto X_0, y \mapsto Y_0$	$X_0 + Y_0 \leq 15$
$5 \rightarrow 6$	$x \mapsto X_0, y \mapsto Y_0$	$X_0 + Y_0 \leq 15$
$6 \rightarrow 7$	$x \mapsto X_0, y \mapsto Y_0 + 10$	$X_0 + Y_0 \leq 15$
$7 \rightarrow 9$	$x \mapsto X_0 - 2, y \mapsto Y_0 + 10$	$X_0 + Y_0 \leq 15$

$9 \rightarrow 11$	$x \mapsto X_0, y \mapsto Y_0 + 10$	$X_0 + Y_0 \leq 15$
$11 \rightarrow 12$	$x \mapsto X_0, y \mapsto Y_0 + 10$	$X_0 + Y_0 \leq 15 \wedge 2 * (X_0 + Y_0 + 10) > 21$
$12 \rightarrow 13$	$x \mapsto 3X_0, y \mapsto Y_0 + 10$	$X_0 + Y_0 \leq 15 \wedge 2 * (X_0 + Y_0 + 10) > 21$
$13 \rightarrow 17$	$x \mapsto 3X_0, y \mapsto 2(Y_0 + 10)$	$X_0 + Y_0 \leq 15 \wedge 2 * (X_0 + Y_0 + 10) > 21$

Path 4 (both "if" are false):

Edge	Symbolic State (PV)	Path Condition (PC)
$1 \rightarrow 2$	$x \mapsto X_0, y \mapsto Y_0$	true
$2 \rightarrow 5$	$x \mapsto X_0, y \mapsto Y_0$	$X_0 + Y_0 \leq 15$
$5 \rightarrow 6$	$x \mapsto X_0, y \mapsto Y_0$	$X_0 + Y_0 \leq 15$
$6 \rightarrow 7$	$x \mapsto X_0, y \mapsto Y_0 + 10$	$X_0 + Y_0 \leq 15$
$7 \rightarrow 9$	$x \mapsto X_0 - 2, y \mapsto Y_0 + 10$	$X_0 + Y_0 \leq 15$
$9 \rightarrow 11$	$x \mapsto X_0, y \mapsto Y_0 + 10$	$X_0 + Y_0 \leq 15$
$11 \rightarrow 14$	$x \mapsto X_0, y \mapsto Y_0 + 10$	$X_0 + Y_0 \leq 15 \wedge 2 * (X_0 + Y_0 + 10) \leq 21$
$14 \rightarrow 15$	$x \mapsto X_0, y \mapsto Y_0 + 10$	$X_0 + Y_0 \leq 15 \wedge 2 * (X_0 + Y_0 + 10) \leq 21$
$15 \rightarrow 16$	$x \mapsto 4X_0, y \mapsto Y_0 + 10$	$X_0 + Y_0 \leq 15 \wedge 2 * (X_0 + Y_0 + 10) \leq 21$
$16 \rightarrow 17$	$x \mapsto 4X_0, y \mapsto 3(Y_0 + 10) + 4X_0$	$X_0 + Y_0 \leq 15 \wedge 2 * (X_0 + Y_0 + 10) \leq 21$

(c)
Path 1 is feasible. For example, $X_0 = 20, Y_0 = 10$.
Path 2 is not feasible.
Path 3 is feasible. For example, $X_0 = 2, Y_0 = 3$.
Path 4 is feasible. For example, $X_0 = 0, Y_0 = 0$.

Q2

(a)
Since we want at most one of the Boolean variables of a_1, a_2, a_3 and a_4 is true, any two variables cannot be true. Then we have:

$$\neg((a_1 \wedge a_2) \vee (a_1 \wedge a_3) \vee (a_1 \wedge a_4) \vee (a_2 \wedge a_3) \vee (a_2 \wedge a_4) \vee (a_3 \wedge a_4))$$

Apply De Morgan's laws to eliminate the negation outside the parentheses:

$$(\neg(a_1 \wedge a_2) \wedge \neg(a_1 \wedge a_3) \wedge \neg(a_1 \wedge a_4) \wedge \neg(a_2 \wedge a_3) \wedge \neg(a_2 \wedge a_4) \wedge \neg(a_3 \wedge a_4))$$

Apply De Morgan's laws to transfer into an equivalent CNF:

$$(\neg a_1 \vee \neg a_2) \wedge (\neg a_1 \vee \neg a_3) \wedge (\neg a_1 \vee \neg a_4) \wedge (\neg a_2 \vee \neg a_3) \wedge (\neg a_2 \vee \neg a_4) \wedge (\neg a_3 \vee \neg a_4)$$

(b)
The sentence is valid. We can prove it by two steps:

1. LHS Implies RHS:
   If $\forall x \cdot \exists y \cdot P(x) \vee Q(y)$ is true, then for each x in the domain, either P(x) is true for that particular x, or there exists some y for which Q(y) is true.
   - If P(x) is true for every x, then $\forall$x·P(x) is true, and thus the RHS is true.
   - If P(x) is not true for some x, then according to the LHS, $\exists$y·Q(y) must be true for the LHS to hold. This also makes the RHS true.

   Therefore, in all cases where the LHS is true, the RHS is also true.
2. RHS Implies LHS:
   If $(\forall x \cdot P(x)) \vee (\exists y \cdot Q(y))$ is true, then we have two cases:

- If ∀x·P(x) is true, then for every x, P(x) is true. Therefore, $\exists y \cdot P(x) \vee Q(y)$ is trivially true for any y, making the LHS true.
- If ∃y·Q(y) is true, then there exists at least one yy such that Q(y) is true. This implies that for every x, $P(x) \vee Q(y)$ will be true (since Q(y) is true for some y). Thus, the LHS is again true.

Therefore, in all cases where the RHS is true, the LHS is also true.
Since both the LHS implies the RHS and the RHS implies the LHS in all possible models, the statement is valid.

(c)
We can prove that the FOL sentence is invalid.
Prove by constructing a model M of the RHS that is not a model of the LHS.

$$S = \{0,1\}, M(P) = \{(0,0)\}, M(Q) = \{(1,1)\}$$

It means that P(0,0) is true while all others are false; Q(1,1) is true while all others are false.
Then we can evaluate both LHS and RHS.
LHS: $(\forall x \cdot \exists y \cdot P(x,y) \vee Q(x,y))$
For x = 0, we have y = 0 to make P(x,y) true.
For x = 1, we have y = 1 to make Q(x,y) true.
So LHS is true because for every x, we can find a y to make $P(x,y) \vee Q(x,y)$ true.
RHS: $(\forall x \cdot \exists y \cdot P(x,y)) \vee (\forall x \cdot \exists y \cdot Q(x,y))$
For x = 0, we have y = 0 to make P(x,y) true.
For x = 1, we don't have y to make P(x,y) true. Because only P(0,0) is true.
Therefore, $(\forall x \cdot \exists y \cdot P(x,y))$ is false.
For x = 0, we don't have y to make Q(x,y) true. Because only Q(1,1) is true.
For x = 1, we have y = 1 to make Q(x,y) true.
Therefore, $(\forall x \cdot \exists y \cdot Q(x,y))$ is false.
Thus, RHS is false.
Since LHS is true while RHS is false in this model, the FOL sentence is invalid.

(d)

- (a)
  It satisfies the formula Φ.
  To satisfy $\exists x \exists y \exists z\ (P(x,y) \wedge P(z,y) \wedge P(x,z) \wedge \neg P(z,x))$ in M_1, we should make:
  $$x < y, z < y, x < z, z \geq x$$
  This can be satisfied since $x < z\ and\ x \leq z$ can be satisfied when $x < z$.
- (b)
  It violates the formula Φ.
  To satisfy $\exists x \exists y \exists z\ (P(x,y) \wedge P(z,y) \wedge P(x,z) \wedge \neg P(z,x))$ in M_2, we should make:
  $$y = x + 1, y = z + 1, z = x + 1$$
  Based on the equations $y = z + 1, z = x + 1$, we can get that $y = x + 2$. It leads to a contradiction since $y = x + 1$.
- (c)
  It satisfies the formula Φ.
  To satisfy $\exists x \exists y \exists z\ (P(x,y) \wedge P(z,y) \wedge P(x,z) \wedge \neg P(z,x))$ in M_3, we should make:
  $$x \subseteq y, z \subseteq y, x \subseteq z, not\ z \subseteq x$$

It says that x is a subset of y, z is a subset of y, x is a subset of z and z is not a subset of x. This can be satisfied. For example, x = {1}, y = {1,2,3}, z = {1,2}. Then the statements $x \subseteq y, z \subseteq y, x \subseteq z, not\ z \subseteq x$ is true. Thus it satisfies the formula Φ.

(e)

Klieber et al. [1] shows an efficient CNF encoding for the constraint at-most-one. aims to reduce the number of required CNF clauses from $O(n^2)$ to $O(n)$, using the commander-variable method. First, the set of n variables can be divided into m groups $(G_1, G_2, \dots, G_m)$. Each group G_i is assigned a commander variable c_i. The commander variable is true if at least one variable in its group is true. Then we can put the following constraints to apply at-most-one:

Constraint 1: at most one variable is true.

$$\bigwedge\nolimits_{x_j \in G_i} \bigwedge\nolimits_{x_k \in G_i, k<j} (\neg x_j \vee \neg x_k)$$

Constraint 2: if a commander variable of a group is false then every variable of the group is false.

$$\bigwedge\nolimits_{x_j \in G_j} (c_i \vee \neg x_j)$$

Constraint 3: Exactly one command variable in all the groups is true.

$$(c_1 \vee c_2 \vee \dots \vee c_m) \wedge \bigwedge\nolimits_{i<m} \bigwedge\nolimits_{j<i} (\neg c_i \vee \neg c_j)$$

Grouping Variables into Hierarchies: The variables are divided into groups, each containing k variables. For instance, if k=2, the grouping forms a binary tree structure. In the general case, the total number of groups formed is given by the formula:

$$\sum_{i=1}^{\infty} \left(\frac{n}{k}\right) = \left(\frac{n}{k}\right) + \left(\frac{n}{k^2}\right) + \left(\frac{n}{k^3}\right) + \cdots = \left(\frac{\frac{1}{k}}{1-\frac{1}{k}}\right) n = \left(\frac{n}{k-1}\right)$$

When n is the number of variables of the group, the first constraint needs $n(n-1)/2$ clauses. The second needs n clauses. Therefore, based on the constraints, each group needs $\left(\frac{k(k+1)}{2}\right)$ clauses. Multiplying the number of clauses per group by the total number of groups gives the total number of clauses for the whole hierarchy. The optimal group size for minimizing the total number of clauses is found to be $k = 3$. With $k = 3$, the formula for the total number of clauses becomes $3n$, which is linear in n. Thus, it reduces the number of clauses needed from $O(n^2)$ to $O(n)$.

Q3

(a)

For a magic square of size n×n, the magic constant (the common sum along any row, column, or diagonal) is given by $\frac{n(n^2+1)}{2}$.

Val/2: the value in the cell at row i and column j.

Int/1: the number is an integer.

$$Int(n) \wedge n \geq 1$$

$$\bigwedge_{\substack{1 \leq i \leq n, 1 \leq j \leq n, Int(i), Int(j) \\ 1 \leq r \leq n, 1 \leq c \leq n, Int(r), Int(c) \\ i \neq r \vee j \neq c}} Val(i,j) \neq Val(r,c)$$

$$\bigwedge_{1 \leq i \leq n, 1 \leq j \leq n, Int(i), Int(j)} 1 \leq Val(i,j) \leq n^2$$

$$\bigwedge_{1\le i\le n, Int(i)} \sum_{j=1}^{n} Val(i,j) = \frac{n(n^2+1)}{2}$$

$$\bigwedge_{1\le j\le n, Int(j)} \sum_{i=1}^{n} Val(i,j) = \frac{n(n^2+1)}{2}$$

$$\sum_{1\le i\le n, Int(i), i=j} Val(i,j) = \frac{n(n^2+1)}{2}$$

$$\sum_{1\le i\le n, Int(i), j=n-i+1} Val(i,j) = \frac{n(n^2+1)}{2}$$

Q4

(e)

The program is provided as below, with 1210 paths:

```
  def test_eighteen(self):
      prg1 = "havoc x; while x > 0 and x < 20 do x:= x - 1; havoc y; while y > 0 and y < 20 do y:= y - 1; havoc z; while
z > 0 and z < 10 do z:= z - 1"
      ast1 = ast.parse_string(prg1)
      engine = sym.SymExec()
      st = sym.SymState()
      out = [s for s in engine.run(ast1, st)]
      self.assertEquals(len(out), 1210)
```

REFERENCE

[1] Klieber W, Kwon G. Efficient CNF encoding for selecting 1 from n objects[C]//Proc. International Workshop on Constraints in Formal Verification. 2007: 14.