Written + Programming Assignment

Kshitij Srivastava
M718099

Theory

Ques-1.

(a) $\exists x \exists y [muffin(x) \wedge chocolate(x) \wedge muffin(y) \wedge chocolate(y) \wedge x \neq y] \wedge \forall z [muffin(z) \wedge chocolate(z) \rightarrow (z=x \vee z=y)]$

(b) $\neg \exists x \forall y [cat(x) \wedge smart\_rat(y) \wedge likes(x,y)]$

$likes(x,y)$: x likes y

(c) $E(x)$: Class average in English
$M(x)$: Class average in Maths.
$\forall x [E(x) \wedge M(x) \longrightarrow E(x) > M(x)]$

(d) $\forall x [Diamond(x) \vee platinum(x)] \longrightarrow precious(x)$

(e) $submit(x)$: submit assignments in course X
$pass(x)$: pass with good grades in course X

~~$\forall x [\neg S(x) \rightarrow (S(x)$~~

$\forall x [\neg submit(x) \longrightarrow (submit(x) \rightarrow pass(x))]$

(f) $angry(x)$: x is angry
$hungry(x)$: x is hungry
$shark(x)$: x is a shark
$whale(x)$: x is a whale

$\forall x [(shark(x) \vee whale(x)) \wedge (angry(x) \vee hungry(x)) \longrightarrow attack(x)]$

$attack(x)$: x attacks.

(g) $\forall x [fish(x) \longrightarrow swim(x)]$

(h) ~~$Eat(x,t)$: Mary can eat x at time t.~~
~~$\exists x \forall t Eat(x,t) \wedge \forall x \exists t Eat(x,t) \wedge \neg \exists x \forall t$~~
$Mary(x)$: x is Mary
$Eat(x,y,t)$: x can eat y at time t
$\exists y \forall t Eat(x,y,t)$

(h) Eat(y,t) : Mary can eat candy y at time t

$\exists y \forall t\ Eat(y,t) \wedge \forall y \exists t\ Eat(y,t) \wedge \forall y \forall t \rightarrow Eat(y,t)$

(i) fan(x) : x is a fan
superhero(x) : x is a superhero
likes(x,y) : x likes y
killer(x) : x is a killer.

$\exists x\ fan(x) \wedge \forall y\ superhero(y) \wedge likes(x,y) \longrightarrow \neg killer(y)$

(j) shopkeeper(x) : x is a shopkeeper
customer(x) : x is a customer
member(x) : x is a member
sells(x,y) : x sells to y

$\exists y\ shopkeeper(y) \wedge \forall x\ customer(x) \wedge member(x) \rightarrow sells(y,x)$

Ques-2.

Init (Object(Q) ∧ Object(X) ∧ V-armAt(Top-left) ∧ H-armAt(Bottom-right)

Goal (Object(Q) ∧ ¬ Object(X))

Actions :

(1) Action Move-Varm-Vertical (V-arm, vertical):
precondition: V-arm-At Row(r) ∧ Row(r) ∧ Arm(a)
effect : ¬ V-arm-At Row(r)

(2) Action Move-H-arm-Horizontal (H-arm, horizontal):

---

(1) Action Move-Arm (arm, direction):
precondition : ArmAt(c,r) ∧ Row(r) ∧ column(c)
effect : ¬ ArmAt(c,r)

(2) Action Move_V_arm_Horizontal (V_arm, horizontal):

Precondition: AtColumn(c) ∧ No of objects In GoalPosition(l) ∧ objectAtGoalPosition(o) ∧ Object(o)

Effect: → Atcolumn(c)

(3) Action V_arm_burst_object (V_arm, object):

precondition: ObjectAt(c, r) ∧ V_armAt(c, r) ∧ Empty(V_arm)

effect: → objectAt(c, r) ∧ V_armAt(c, r) ∧ Empty(V_arm)

(4) Action H_arm_move_vertical (H_arm, vertical):

precondition: V_arm_obstructedAt(c, r) ∧ Row(r)

effect: → V_arm_obstructedAt(c, r) ∧ H_arm_At(r)

(5) Action pick_object (arm, object):

precondition: objectAt(c, r) ∧ ArmAt(c, r) ∧ Empty(arm) ∧ object(o)

effect: ArmAt(c, r) ∧ → Empty(arm)

(6) Action drop_object (arm, object):

precondition: ArmAt(c, r) ∧ EmptyAt(c, r) ∧ Object(o) ∧ Arm(arm)

effect: ArmAt(c, r) ∧ → EmptyAt(c, r)

Ques-3

Init (Organising_Head (Kshitij) ∧ SeminarDate (date) ∧
Organising Committee (committee) ∧ Lecture (Professor X) ∧
Venue (C102))

Actions:

(1) Action : Take-permission (department, for):
precondition : NoPermission (department, for) ∧ Department (department)
effect : → NoPermission (department, for)

(2) Action Invite (profesor X):
precondition: professorAvailableAt (date)
effect : professorInvitedAt (date)

(3) Action arrangefunds (funds, from):
precondition: Need (funds) ∧ department (Accounts)
effect : → Need (funds)

(4) Action make-announcement (about, by, to):
precondition: professorInvited (Professor X) ∧ Venue (C102) ∧
fundsArranged (funds)
effect : made-announcement (seminar, mail, students)

(5) Action Lunch (professors):
precondition: professorPresent (professors) ∧ professorshungry (prof)
∧ food Available (food)
effect: → professorHungry (prof)

(6) Action getInternet (Venue, IT):

precondition: checkInternet (~~IES~~ NO) ∧ Venue (C-102) ∧ department (IT)

effect: Check_Internet (YES)

(7) Action getEquipments (Venue, FMS):

precondition: department (FMS) ∧ NoEquipments At (venue)

effect: ¬ NoEquipments At (venue)

(8) Action CleanVenue (venue, FMS):

precondition: NoCleanVenue (venue) ∧ department (FMS)

effect: ¬ NoCleanVenue (venue)

(9) Action fixTiming (date, Academics)

precondition: NoScheduleSeminar (date) ∧ department (Academics)

effect: ¬ NoScheduleSeminar (date)

Query.

P(x) : x is a paratha

C(x) : x is Chenopodium

R(x) : x is Radish

Eat(x) : I eat x

like(x) : I like x

hungry(x) : I sleep hungry

~~(1) $\forall x$ P(x) $\rightarrow$ (C(x) $\leftrightarrow$ R(x))~~

(1) $\forall x$ [P(x) $\rightarrow$ (C(x) $\leftrightarrow$ $\neg$ R(x))]

(2) $\forall x$ [P(x) $\wedge$ like(x) $\rightarrow$ eat(x)]

(3)(a) $\forall x$ [(C(x) $\leftrightarrow$ $\neg$ R(x)) $\rightarrow$ $\neg$ like(x)]

(b) $\neg$ like(x) $\rightarrow$ $\neg$ eat(x)

(4) $\neg$ eat(x)

(5) $\neg$ eat(x) $\rightarrow$ hungry(x)

Assuming, all predicates have been universally instantiated.

[(C(x) $\leftrightarrow$ $\neg$ R(x))] as X.

steps	Formula	Derivation
1.	$\neg$ P $\vee$ X	given
2.	$\neg$ [P $\wedge$ like] $\vee$ E	given
3.	X $\rightarrow$ $\neg$ Like	given
4.	like $\rightarrow$ $\neg$ E	given
5.	eat	negated conclusion
6.	$\neg$ P $\vee$ $\neg$ Like	1, 3 (Resolution)
7.	$\neg$ P	2, 4 [Resolution: (2) $\neg$ [P $\wedge$ like] $\vee$ E, (4) like $\vee$ $\neg$ E] $\rightarrow$ $\neg$ [P $\wedge$ like] $\vee$ like
8.	$\neg$ Like	6, 7
9.	$\neg$ E	4, 8
10.	Empty clause.	5, 9 $\rightarrow$ Indicates Contradiction.

- The empty clause contradicts our assumption, i.e., I didn't eat anything.
- Hence, I will sleep hungry. (from ⑤)