Rohit Ravishankar

1)

$\neg$ Rainy $\rightarrow$ Pleasant

Rainy $\rightarrow$ cloudy $\wedge$ $\neg$ Pleasant

Pleasant $\vee$ Cloudy $\rightarrow$ Hiking

Hiking $\rightarrow$ Happy

Step 2 Replacing "$\rightarrow$"

$\neg$ ($\neg$ Rainy) $\vee$ Pleasant

$\neg$ Rainy $\vee$ (cloudy $\wedge$ $\neg$ Pleasant)

$\neg$ (Pleasant $\vee$ cloudy) $\vee$ Hiking

$\neg$ Hiking $\vee$ Happy

Step 3

Rainy $\vee$ Pleasant

$\neg$ Rainy $\vee$ (cloudy $\wedge$ $\neg$ Pleasant)

($\neg$ Pleasant $\wedge$ $\neg$ Cloudy) $\vee$ Hiking

$\neg$ Hiking $\vee$ Happy

$\Downarrow$

1. $R \vee P$
2. $\neg R \vee (C \wedge \neg P)$
3. $(\neg P \wedge \neg C) \vee G$
4. $\neg G \vee H$

we prove by contradiction, by asserting the clauses of premise & the negation of the conclusion

1. $R \vee P$
2. $\neg R \vee C$
3. $\neg R \vee \neg P$
4. $G \vee \neg P$
5. $G \vee \neg C$
6. $\neg G \vee H$

Premise & by resolution of the statements on the previous side

a) Let ~~today~~ not be Rainy, i.e., $\neg R$

7. $\Rightarrow P$ adding $\neg R$ to 1
8. $\Rightarrow \neg R$ adding 7 to 2

Hence, today is not rainy and we cannot prove by contradiction.

b) Let "not Happy", i.e., $\neg H$

13. $\Rightarrow H$ Adding 12 to 6

Hence, our assumption contradicts 13 and by proof by contradiction I am happy

c) Let today not be good for hiking, i.e., $\neg G$

9. $\neg P$ adding $\neg G$ to 4
10. R adding 9 to 1
11. C adding 10 to 2
12. G adding 11 to 5

Our assumption contradicts 12 and hence, by proof by contradiction, today is good for hiking

2//

1. $O \rightarrow W$

$\Rightarrow \neg O \vee W$

2. $O \rightarrow O$

$\Rightarrow \neg O \vee O$

$\Rightarrow$ True

3. $(O \rightarrow W) \rightarrow (\neg O \rightarrow \neg W)$

$\Rightarrow \neg (O \rightarrow W) \vee (\neg O \rightarrow \neg W)$

$\Rightarrow \neg (\neg O \vee W) \vee (\neg(\neg O) \vee \neg W)$

$\Rightarrow (O \wedge \neg W) \vee (O \vee \neg W)$

4. $O \vee W \vee (O \rightarrow W)$

$\Rightarrow O \vee W \vee (\neg O \vee W)$

$\Rightarrow$ True

5. $(O \rightarrow W) \rightarrow ((O \wedge C) \rightarrow W)$

$\Rightarrow \neg (O \rightarrow W) \vee ((O \wedge C) \rightarrow W)$

$\Rightarrow \neg (\neg O \vee W) \vee (\neg (O \wedge C) \vee W)$

$\Rightarrow (O \wedge \neg W) \vee (\neg O \vee \neg C \vee W)$

6. $(W \rightarrow C) \wedge (C \rightarrow \neg O) \wedge \neg (O \rightarrow W)$

$\Rightarrow (\neg W \vee C) \wedge (\neg C \vee \neg O) \wedge (O \wedge \neg W)$

C	O	W	① $O \rightarrow W$	② $O \rightarrow O$	③ $(O \rightarrow W) \rightarrow (\neg O \rightarrow \neg W)$	④ $O \vee W \vee (O \rightarrow W)$	⑤ $(O \rightarrow W) \rightarrow ((O \wedge C) \rightarrow W)$	⑥ $(W \rightarrow C) \wedge (C \rightarrow \neg O) \wedge \neg (O \rightarrow W)$
0	0	0	1	1	1	1	1	0
0	0	1	1	1	0	1	1	0
0	1	0	0	1	1	1	1	0
0	1	1	1	1	1	1	1	0
1	0	0	1	1	1	1	1	0
1	0	1	1	1	0	1	1	0
1	1	0	0	1	1	1	1	0
1	1	1	1	1	1	1	1	0

③ & ① are satisfiable statements

④, ② & ⑤ are valid statements [All truth-table values for the sentences are true]

⑥ - unsatisfiable [All truth-table values are false]

Rohit Ravishankar

③ is not valid when old is false & wise is true

① is not satisfiable / not valid when old is true & wise is false

Rohit Ravishankar

3/

1.

Cats(x) : x is a cat

Animal(x) : x is a animal

$\forall x : cats(x) \rightarrow Animal(x)$

2.

$\exists x\ \exists y\ Animal(x) \wedge Afraid(x, y) \rightarrow Run(x, y) \vee Hide(x, y)$

3.

$\exists x\ \exists y\ cat(x) \wedge Car(y) \rightarrow Afraid(x, y)$

$\exists x\ \exists y\ Cat(x) \wedge Dog(y) \rightarrow Afraid(x, y)$

4.

Cat (Louie)

5.

Dog (Jake)

6.

$\exists x\ \exists y\ Animal(x) \wedge Hide(x, y) \rightarrow \neg Seen(x)$

7.

Seen (Louie)

CNF form corresponding to the rules

1. $\neg Cats(x) \vee Animal(x)$
2. $\neg Animal(x) \vee \neg Afraid(x,y) \vee Run(x,y) \vee Hide(x,y)$
3. $\neg Cat(x) \vee \neg Car(y) \vee Afraid(x,y)$
4. $\neg Cat(x) \vee \neg Dog(y) \vee Afraid(x,y)$
5. $Cat(Louie)$
6. $Dog(Jake)$
7. $\neg Animal(x) \vee \neg Hide(x,y) \vee \neg Seen(x)$
8. $Seen(Louie)$

Proof by contrdiction

Assumption: $\neg Run(Louie, jake)$

→ Substitute for in 2

$\neg Animal(Louie) \vee \neg Afraid(Louie, jake) \vee Hide(Louie, jake) \rightarrow 9$

→ Substitute 9 in 7

$\neg Animal(Louie) \vee \neg Seen(Louie) \vee \neg Afraid(Louie, jake) \rightarrow 10$

→ Substitute 10 in 4

$\neg Animal(Louie) \vee \neg Seen(Louie) \vee \neg Cat(Louie) \vee \neg Dog(Jake) \rightarrow 11$

→ Substitute 11 in 5

$\neg Animal(Louie) \vee \neg Seen(Louie) \vee \neg Dog(jake) \rightarrow 12$

→ Substitute 12 in 6

$\neg Animal(Louie) \vee \neg Seen(Louie)$

→ Substitue 13 in 8

$\neg Animal(Louie) \rightarrow 14$

→ Substitute 14 in 1

$\neg$ Cat (Louie) → 15

However, in 5 we get Cat (Louie).
Since, 5 and 15 are a contradiction, it can be proved that Louie is running from jake

4.

Desktop — gprolog — 80×24

```
| ?- setof(X, grandchild(X, elizabeth), List).

List = [beautrice,eugenie,harry,james,louise,peter,william,zara]

yes
| ?-  setof(X, brotherinlaw(X, diana), List).

no
| ?- setof(X, greatgrandparent(X, zara), List).

List = [george,mum]

yes
| ?- setof(X, ancestor(X, eugenie), List).

List = [andrew,elizabeth,george,mum,philip,sarah]

yes
| ?- setof(X, ancestor(X, eugenie), List).

List = [andrew,elizabeth,george,mum,philip,sarah]

yes
| ?- 
```

An interesting point to note is brother-in-law of Diana is “no” because we follow the simple definition of sibling-in-law as stated in the question. A brother-in-law or sister-in-law is defined as sisters spouse or brothers spouse. However, since Diana has no brother-in-laws as per our definition, “no” is returned.

5)

a)

(i) Action (Move (x, y, r),

Precondition: Robot(rob) $\wedge$ At(rob, x, r) $\wedge$ Room(r) $\wedge$ Location(x, r) $\wedge$ Location(y, r)

Effect: $\neg$At(rob~~ot~~, x, r) $\wedge$ At(rob~~ot~~, y, r))

(ii) Action (Push(b, x, y, r),

Precondition: Box(b) $\wedge$ Room(r) $\wedge$ At(b, x, r) $\wedge$ Robot(rob) $\wedge$ Location(x, r) $\wedge$ Location(y, r)

Effect: $\neg$At(b, x, r) $\wedge$ At(b, y, r))

(iii) Action (TurnOn(s),

Precondition: Switch(s) $\wedge$ Room(r) $\wedge$ At(s, r) $\wedge$ $\neg$On(s) $\wedge$ Robot(rob) $\wedge$ At(rob, r)

Effect: $\neg$Off(s))

(iv) Action (TurnOff(s),

Precondition: Switch(s) $\wedge$ Room(r) $\wedge$ At(s, r) On(s) $\wedge$ Robot(rob) $\wedge$ At(rob, r)

Effect: Off(s))

b)

Init (Robot(rob) $\wedge$ Room(R_1) $\wedge$ Room(R_2) $\wedge$ Room(R_3) $\wedge$ Box(B1) $\wedge$ Box(B_2) $\wedge$ Box(B_3) $\wedge$ Room(Hallway) $\wedge$ At(rob, R_2) $\wedge$ At(B_1, R_1) $\wedge$ At(B_2, R_1) $\wedge$ At(B_3, R_3))

c//

Goal (At(B_1, R_3):

Action (Move (start, D_2, R_2),
Precondition: Robot(rob) ∧ Room(R_2) ∧ At(rob, Start, R_2) ∧ Location (start, R_2) ∧ Location (D_2, R_2)
Effect: ¬ At (rob, start, R_2) ∧ At (rob, D_2, R_2)).

Action (move (D_2, D_1, Hallway),
Precondition: Robot (rob) ∧ Room (Hallway) ∧ At(rob, D_2, Hallway) ∧ Location (D_2, Hallway) ∧ Location (D_1, Hallway)
Effect: ¬ At (rob, D_2, Hallway) ∧ At (rob, D_1, Hallway))

Action (move (D_1, x_1, R_1),
Precondition: Robot (rob) ∧ Room (R_1) ∧ At(rob, D_1, R_1) ∧ Location (D_1, R_1) ∧ Location(x_1, R_1))
Effect: ¬ At (rob, D_1, R_1) ∧ At (rob, x_1, R_1))

Action (Push (B_1, x_1, D_1, R_1),
Precondition: Robot (rob) ∧ Room (R_1) ∧ At (B_1, x_1, R_1) ∧ Box (B_1) ∧ Location (x_1, R_1) ∧ Location (D_1, R_1))
Effect: ¬ At (B_1, x_1, R_1) ∧ At (B_1, D_1, R_1))

Action (Push (B_1, D_1, D_3, Hallway),
Precondition: Robot (rob) ∧ Room (Hallway) ∧ At (B_1, D_1, hallway) ∧ Box(B_1) ∧ Location (D_1, Hallway) ∧ Location (D_3, Hallway)
Effect: ¬ At (B_1, D_1, Hallway) ∧ At (B_1, D_3, Hallway))

Action (Push (B_1, D_3, end, R_3),
Precondition: Box (B_1) ∧ Robot(rob) ∧ Room(R_3) ∧ At (B_1, D_3, R_3) ∧ Location (D3, R_3) ∧ Location (end, R_3)
Effect: ¬ At (B_1, D3, R_3) ∧ At (B_1, end, R_3)))

→ We represent the rooms using the notation of 'R' subscripted by room numbers. Eg $\{R_1, R_2, R_3\}$

→ We represent the boxes using the notation of 'B' subscripted by box numbers. Eg $\{B_1, B_2, B_3\}$

→ We represent the doors to the rooms as 'D' subscripted by the room number, i.e., D_1 is the door to R_1 & D_2 to R_2 and D_3 to R_3.

→ In the first step, the robot moves from the start position in Room R_2 to the position D_2 in R_2. As stated in the question D_2 is in R_2 and the hallway.

→ In the second step, the robot moves from D_2 which is at R_2 and the hallway to D_1 in the hallway again. The constraint is that we can move from position x to position y within the room only.

→ In the 3rd step, the robot moves from position D_1 in the room to the box location in the room.

→ And so, we continue the process of moving the box B_1 from Room R_1 to Room R_3 in a similar fashion

c) The branching factor for this version of the problem could be

$$\underbrace{3 * 3}_{\text{3 boxes \& 3 rooms (2 rooms + hallway)}} + \underbrace{2 * 3}_{\text{3 rooms \& 2 turn off/on action}} + 1*3 = 18$$

$1*3$ → only 1 move/push action per room as there is

If the problem were for k rooms & m boxes the problem could be represented as

$$\boxed{m * k + 2 * k + 1 * k} = \boxed{m * k + 3 * k}$$