Assignments week 5/6 – Chris Pool

4.2.1

¬¬p→p

1 F(¬¬p→p)	✓
2 T(¬p→p)	1,T¬ ✓
3 T¬p	2,F→ ✓
4 Fp	2,F→
5 Fp	3,T¬

This is not valid because we can falsify it

((p→q)→p)→p

	1. F((p→q)→p)→p ✓	
	1,F→ ✓	
	3. Fp, 1,F→	
4. F(p→q), 2,T→ ✓		5. Tp , 2,T→
6. T(p), 4, F→		
7. F(q), 4F→		

Unable to falsify , both branches require P to be true

(¬p→¬q)→(q→p)

	1. F((¬p→¬q)→(q→p)) ✓	
	2. T(¬p→¬q), 1,F→✓	
	3. F (q→p), 1,F→✓	
	4. T(q), 2,F→	
	5. F(p), 2,F→	
6. F(¬p), 4,T→✓		7. T(¬q), 4,T→✓
8. T(p), 6, F¬		9. F(q), 7, T¬

Unable to falsify

p→(p∧(q∨p))

	1. F(p→(p∧(q∨p))) ✓	
	2. T(p), 1,F→	
	3. F(p∧(q∨p)), 1,F→✓	
4. F(p), 3,F∧		5. F(q∨p), 3,F∧ ✓
		6. F(q) , 5,F∨
		7. F(p) , 5,F∨

Unable to falsify, P is once false and once true

Assignments week 5/6 – Chris Pool

(p∨(q∧r))→((p∨q)∧(p∨r))

		1. F((p∨(q∧r))→((p∨q)∧(p∨r)) ✓		
		2. T(p∨(q∧r)) 1,F→ ✓		
		3. F((p∨q)∧(p∨r)) 1,F→ ✓		
4. T(p) , 2,T∨			5. T(q∧r), 2, T∨ ✓	
8. F(p∨q), 3,F∧ ✓	9. F(p∨r), 3,F∧ ✓		6. T(q) , 5, T∧	
10. F(p) , 8, F∨	12. F(p) , 9, F∨		7. T(r), 5, T∧	
11. F(q), 8,F∨	13. F(r) , 9, F∨		14. F(p∨q), 3,F∧ ✓	15. F(p∨r), 3,F∧ ✓
			16. F(p) , 14, F∨	18. F(p) , 15, F∨
			17. F(q), 14,F∨	19. F(r) , 15, F∨

Unable to falsify

4.4.1

¬(¬¬p→p)	Negate
p→p	Convert to NNF
¬p ∨ p	Convert to NNF
[¬p ,p]	CNF

¬ (((p→q)→p)→p)	Negate
((p→q)→p)∧¬p	Convert to NNF
(¬(p→q)∨p)∧¬p	Convert to NNF
(p∧¬q∨p)∧¬p	Convert to NNF
((p∨p)∧(¬q∨p))∧¬p	Convert to CNF
[[p,p],[¬q,p],[¬p]]	CNF

¬((¬p→¬q)→(q→p))	Negate
(¬p→¬q)∧¬(q→p)	Convert to NNF
(¬p→¬q)∧(q∧¬p)	Convert to NNF
(¬¬p∨¬q)∧(q∧¬p)	Convert to NNF
(p∨¬q)∧(q∧¬p)	Convert to CNF
[[p,¬q],[q],[¬p]]	CNF

¬(p→(p∧(q∨p)))	Negate
p ∧ ¬ (p∧(q∨p)))	Convert to NNF
p∧(¬p∨¬(q∨p))	Convert to NNF
p∧(¬p∨(¬q∧¬p))	Convert to NNF
p∧((¬p∨¬q)∧(¬p∨¬p))	Convert to CNF
[[p],[¬p,¬q],[¬p,¬p]]	CNF

¬((p∨(q∧r))→((p∨q)∧(p∨r)))	Negate
(p∨(q∧r))∧(¬((p∨q)∧(p∨r)))	Convert to NNF
(p∨(q∧r))∧(¬(p∨q)∨¬(p∨r))	Convert to NNF
(p∨(q∧r))∧((¬p∧¬q)∨(¬p∧¬r))	Convert to NNF
((p∨q)∧(p∨r))∧((¬p∧¬q)∨(¬p∧¬r))	Convert to CNF
((p∨q)∧(p∨r))∧((¬p∨(¬p∧¬r))∧(¬q∨(¬p∧¬r)))	Convert to CNF
((p∨q)∧(p∨r))∧((¬p∨¬p)∧(¬p∨¬r)∧(¬q∨¬p)∧(¬q∨¬r))	Convert to CNF
[[p,q],[p,r],[¬p,¬p],[¬p,¬r],[¬q,¬p],[¬q,¬r]]	CNF

4.4.5

[¬p ,p]	Set
[¬p ,p]	Resolution using p, not possible to create empty set.

[[p,p],[¬q,p],[¬p]]	CNF
[[p],[¬q,p],[¬p]]	SET
[[],],[¬q,p],[]]	Resolution with p

[[p,¬q],[q],[¬p]]	CNF allready in set
[[],[¬q,p],[]]	Resolution with p

[[p],[¬p,¬q],[¬p,¬p]]	CNF
[[p],[¬p,¬q],[¬p]]	SET
[[],[¬p,¬q],[]]	Resolution with p

[[p,q],[p,r],[¬p,¬p],[¬p,¬r],[¬q,¬p],[¬q,¬r]]	CNF
[[p,q],[p,r],[¬p],[¬p,¬r],[¬q,¬p],[¬q,¬r]]	SET
[[q],[p,r],[],[¬p,¬r],[¬q,¬p],[¬q,¬r]]	Resolution with p

Assignments week 5/6 – Chris Pool

Formulate axioms in first-order logic that describes the properties of the spatial relations used in the image models.

∀x∀y(SUPPORTS(x,y) → TOUCH(x,y))
∀x∀y(TOUCH(x,y) → TOUCH(y,x))
∀x∀y(NEAR(x,y) → NEAR(y,x))
∃x∃y(TOUCH(x,y) -> SUPPORTS(x,y)
∀x∀y(TOUCH(x,y) → ¬PART_OF(x,y)
∀x∀y(PART_OF(x,y) → ¬(SUPPORT(x,y) ∧TOUCH(x,y) ∧NEAR(x,y)))
∀x∀y(NEAR(x,y) → ¬(PART_OF(x,y) ∧TOUCH(x,y)))
∀x∀y(PART_OF(x,y) → ¬PART_OF(y,x))