**Name:** Mohamed Nagy Elkholy

**HW8**

2. **Algorithms for Propositional Logic:**

**TT-entails:**

1. Input: KB and a query.
2. Representation: Boolean expressions representing the knowledge base and the query.
3. Stop Condition: The algorithm stops when it finds a row where the KB is true, and the query is false.
4. Mechanism: This algorithm implements the truth table method and relies on the fact that if KB entails a query, there should be no row where KB is true, and the query is false.

**PL-Resolution:**

1. Input: KB in CNF and a query.
2. Representation: Clauses in CNF.
3. Stop Condition: The algorithm stops when it derives the empty clause (contradiction) or the query.
4. Mechanism: Implements the resolution rule, a sound and complete inference rule for propositional logic.

**PL-FC-entails:**

1. Input: KB and a query.
2. Representation: A set of implications (if-then rules).
3. Stop Condition: The algorithm stops when it derives the query or cannot infer any new information.
4. Mechanism: Implements forward chaining, a process of repeatedly applying production rules to derive new information.

**DPLL-Satisfiable:**

1. Input: A CNF formula.
2. Representation: Clauses in CNF.
3. Stop Condition: The algorithm stops when it finds a satisfying assignment or explores all possibilities.
4. Mechanism: Implements the DPLL algorithm, a backtracking search for satisfiability.

**WalkSAT:**

1. Input: A CNF formula and parameters.
2. Representation: Clauses in CNF.
3. Stop Condition: The algorithm stops when it finds a satisfying assignment or reaches a predefined limit of iterations.
4. Mechanism: Implements a local search with random walk, balancing exploration and exploitation.
5. **Using the inference rules for logic:**

| R1: | P(1) | [Given] |
| --- | --- | --- |
| R2: | W(1)∧W(2)∧W(3) | [Given] |
| R3: | ∀x[P(x)→¬R(x)] | [Given] |
| R4: | ∀x[Q(x)∨R(x)] | [Given] |
| R5: | ∀x[(Q(x)∧W(x))→Z(x)] | [Given] |
| R6: | P(1)→¬R(1) | [Using universal instantiation x=1 in R3] |
| R7: | P(1)→¬R(1) | [Using universal instantiation x=1 in R4] |
| R8: | ¬R(1) | [From R1and 6 Using Modus Ponens] |
| R9: | Q(1) | [From R4 and R8 using Disjunctive Syllogism] |
| R10: | W(1) | [From R2] |
| R11: | (Q(1)∧W(1))→Z(1) | [Using Universal Instantiation x=1 in R5] |
| R12: | Z(1) | [From R9, R10, and R11] |
| R13: | ∃xZ(x) | [From R12 using Existential Generalization] |

1. **Chapter 8, Exercise 4:**

∃x∀y (x=y)

This sentence states that there exists an object (x) such that every other object (y) is identical to that object.

1. **Chapter 8, Exercise 10:**
   * + 1. 1. is syntactically invalid and therefore meaningless
          2. correctly expresses the English sentence
          3. is syntactically valid but does not express the meaning of the English sentence
       2. 1. correctly expresses the English sentence
          2. is syntactically valid but does not express the meaning of the English sentence
          3. Question b is a copy of question c
          4. is syntactically invalid and therefore meaningless
       3. 1. correctly expresses the English sentence
          2. correctly expresses the English sentence
          3. is syntactically valid but does not express the meaning of the English sentence
          4. is syntactically valid but does not express the meaning of the English sentence
       4. 1. correctly expresses the English sentence
          2. correctly expresses the English sentence
          3. is syntactically valid but does not express the meaning of the English sentence
          4. correctly expresses the English sentence
       5. 1. correctly expresses the English sentence
          2. correctly expresses the English sentence
          3. is syntactically valid but does not express the meaning of the English sentence
          4. is syntactically invalid and therefore meaningless
2. **Chapter 8, Exercise 30**

Takes(x, c, s): student x takes course c in semester s

Passes(x, c, s): student x passes course c in semester s

Score(x, c, s): the score that the student x gets in course c in semester s

French: French course

Greek: Greek course

Buys(x, y, z): x buys y from z

Sells(x, y, z): x sells y to z

Shaves(x, y): someone x shaves someone y

Born(x, c): someone x is born in country c

Parent(x, y): x is a parent of y

Citizen(x, c, r): x is a citizen of country c for reason r

Resident(x, c): x is a resident of country c

Fools(x, y, t): person fools person y at time t

Student(x): x is a student

Person(x): x is a person

Man(x): x is a man

Barber(x): x is a barber

Expensive(x): x is expensive

Agent(x): x is an agent

Insured(x): x is insured

Smart(x): x is smart

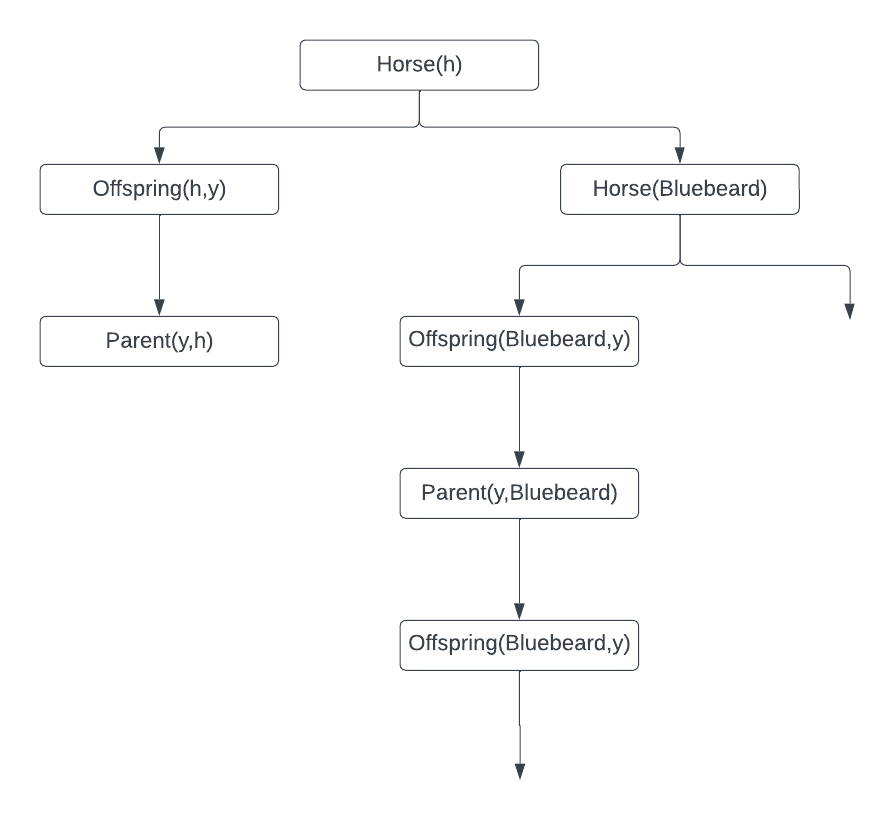
Policy(x): policy is sold by x

Politician(x): x is a politician

* + - 1. ∃x Student(x) ^ Takes(x,French,Spring2001)
      2. ∀x,s Student(x) Takes(x, French, s) → Passes(x, French, s)
      3. ∃x Student(x) ^ Takes(x, Greek, Spring2001) ^ ∀y y≠x → Takes (y, Greek, Spring2001)
      4. ∀s ∃x ∀y Score(x, Greek, s) > Score(y, French, s)
      5. ∀x Person(x) ∧ (∃ y, z Policy(y) ∧ Buys(x, y, z)) → Smart(x)
      6. ∀x, y, z Person(x) ∧ Policy(y) ∧ Expensive(y) → ￢Buys(x, y, z)
      7. ∃x Agent(x) ∧ ∀y, z Policy(y) ∧ Sells(x, y, z) → (Person(z) ∧ ￢Insured(z))
      8. ∃x Barber(x) ∧ ∀y Man(y) ∧ ￢Shaves(y, y) → Shaves(x, y)
      9. ∀x Person(x) ∧ Born(x,UK) ∧ (∀y Parent(y, x) → ((∃r Citizen(y,UK, r)) ∨ Resident(y,UK))) → Citizen(x,UK,Birth).
      10. ∀x Person(x) ∧ ￢Born(x,UK) ∧ (∃y Parent(y, x) ∧ Citizen(y,UK,Birth)) → Citizen(x,UK,Descent)
      11. ∀x Politician(x) → (∃y ∀p Person(y) ∧ Fools(x, y, t)) ∧ (∃ p ∀y Person(y) ⇒ Fools(x, y, p)) ∧￢(∀p ∀y Person(y) → Fools(x, y, p))
      12. ∀x,y,z Person(x) ^ [∃r Citizen(x, Greece, r)] ^ Person(y) ^ [∃r Citizen(y, Greece, r)] ^ Speaks(x,z) → Speaks(y, z)

1. **Axioms in FOL (Adapted from AIMA, first edition):**
2. ∀x ∀y GrandChild(x,y)↔∃z (Child(x,z)^Child(z,y))
3. ∀x ∀y GreatGrandParent(x,y)↔ ∃z,u (Child(y,u)^Child(u,z)^Child(z,x))
4. ∀x ∀y Brother(x,y)↔(Sibling(x,y)^Male(x))
5. ∀x ∀y Sister(x,y)↔(Sibling(x,y)^Female(x))
6. ∀x ∀y Daughter(x,y)↔(Child(x,y)^Female(x))
7. ∀x ∀y Son(x,y)↔(Child(x,y)^Male(x))
8. ∀x ∀y Aunt(x,y)↔∃z (Sibling(x,z)^Female(x)^Child(y,z))
9. ∀x ∀y Uncle(x,y)↔∃z(Sibling(x,z)^Male(x)∧Child(z,y))
10. ∀x ∀y BrotherInLaw(x,y)↔∃z (Spouse(x,z)^Male(x)^Sibling(y,z))
11. ∀x ∀y SisterInLaw(x,y)↔∃z (Spouse(x,z)^Female(x)^Sibling(y,z))
12. ∀x ∀y FirstCousin(x,y)↔∃z,u (Chlid(x,z)^Chlid(y,u)^Sibling(z,u))
13. ∀n ∀p1 ∀p2 Nth-cousin(n,p1,p2)↔∃x,y (Nth-cousin(n-1,x,y)^Child(p1,x)^Child(p2,y))
14. **Chapter 9, Exercise 3:**
15. Not legitimate result of applying Existential Instantiation as it uses the same object.
16. Legitimate result of applying Existential Instantiation.
17. Legitimate result of applying Existential Instantiation.
18. **Chapter 9, Exercise 4:**
    * + 1. {x/A,y/B,z/B}
        2. {y/G(A,B),x/A}
        3. {x/John,y/John}
        4. No unifier exists

1. **Chapter 9, Exercise 7:**
2. ∀x (Horse(x)∨Cow(x)∨Pig(x))→Mammal(x)
3. ∀x ∀y (Offspring(x,y)∧Horse(y))→Horse(x)
4. Horse(Bluebeard)
5. Offspring(Charlie,Bluebeard)
6. ∀x∀y (Offspring(x,y)→Parent(y,x)^Parent(x,y)→Offspring(y,x))
7. ∀x (Mammal(x)→Parent(G(x),x))
8. **Chapter 9, Exercise 16:**



1. “∀x ∀y (Offspring(x,y)∧Horse(y))→Horse(x)” leads to an infinite loop.
2. One
3. **First-Order Logic:**

**Predicates:**

R(x,y): "x rides y"

C(x): "x is a rough character"

H(y): "y is a Harley"

W(y): "y is a BMW"

B(x): "x is a biker"

Y(x): "x is a yuppie"

L(x): "x is a lawyer"

N(x): "x is a nice girl”

D(x,y): "x dates y"

**Axioms in First-Order Logic:**

1. Anyone who rides any Harley is a rough character.

∀ x ((∃ y (H(y) ∧ R(x,y))) → C(x))

2. Every biker rides [something that is] either a Harley or a BMW.

∀ x (B(x) → ∃ y ((H(y) ∨ B(y)) ∧ R(x,y)))

3. Anyone who rides any BMW is a yuppie.

∀ x ∀ y (R(x,y) ∧ W(y) → Y(x))

4. Every yuppie is a lawyer.

∀ x (Y(x) → L(x))

5. Any nice girl does not date anyone who is a rough character.

∀ x ∀ y (N(x) ∧ R(y) → ¬ D(x,y))

6. Mary is a nice girl, and John is a biker.

N(Mary) ∧ B(John)

7. (Conclusion) If John is not a lawyer, then Mary does not date John.

¬ L(John) → ¬ D(Mary,John)

**Transform to CNF:**

1. ∀x ∀y(¬ H(y) ∨ ¬ R(x,y) ∨ C(x))
2. ∀x (¬ B(x) ∨ H(x) ∨ ∃y (¬ R(x,y) ∨ H(y)))
3. ∀x ∀y (H(y) ∨ ¬ R(x,y) ∨ Y(x))
4. ∀x (¬ Y(x) ∨ L(x))
5. ∀x ∃y (¬ N(x) ∨ ¬ C(y) ∨ D(x,y))
6. N(Mary) ∧ B(John)
7. ¬ L(John) ∨ ¬ D(Mary,John)

**Establishing the Conclusion:**

R1: ∀x ∀y(¬ H(y) ∨ ¬ R(x,y) ∨ C(x))

R2: ∀x (¬ B(x) ∨ H(x) ∨ ∃y (¬ R(x,y) ∨ H(y)))

R3: ∀x ∀y (H(y) ∨ ¬ R(x,y) ∨ Y(x))

R4: ∀x (¬ Y(x) ∨ L(x))

R5: ∀x ∃y (¬ N(x) ∨ ¬ C(y) ∨ D(x,y))

R6: N(Mary) ∧ B(John)

R7: ¬ L(John) ∨ ¬ D(Mary,John)

R8: N(Mary) [From R6]

R9: B(John) [From R6]

R10: ¬ Y(John) [Negation of R3]

R11: ¬ R(John,y) ∨ H(y) [Negation of R2]

R12: ¬L(John) [Resolution R7, R10]

R13: ¬R(John,y)∨H(y) [Resolution R12, R11]

The resolution did not lead to acontradiction. Therefore, the conclusion ¬L(John)∨¬D(Mary,John) is established using refutation resolution.