CSCI 377 Midterm 1

Closed book, closed notes

Monday, Sept. 24, 2018

1 True/False (2 pts each, 20 pts total)

(a) True or False? $B \wedge P \models P$ is in the language $\mathcal{L}_{PL}(\Sigma)$, where $\Sigma = \{B, P, F\}$.

(b) True or False? Let $RC(S)$ be the resolution closure of a set S of clauses. Then $RC(RC(S)) = RC(S)$.

(c) True or False? $\forall x \exists y (x \Rightarrow y)$ is a sentence in first order logic.

(d) True or False? $\alpha \vee (\beta \wedge \gamma) \equiv (\alpha \vee \beta) \wedge (\gamma \vee \alpha)$ for any sentences $\alpha, \beta, \gamma \in \mathcal{L}_{PL}(\Sigma)$.

(e) True or False? If I generate a random 3-CNF sentence with 100 clauses over 100 variables, the probability is greater than 50% that it will be satisfiable.

(f) True or False? For sentences $\alpha, \beta \in \mathcal{L}_{PL}(\Sigma)$, $\alpha \models \beta$ iff $\alpha \wedge \neg\beta$ is satisfiable.

(g) True or False? Suppose you have a conjunction of 5 CNF clauses, each of which mentions exactly three different variables (either positive or negative literals). This conjunction MUST be satisfiable.

(h) True or False? There are exactly 2^n unique truth tables over n Boolean variables.

(i) True or False? $(A \Leftrightarrow (B \wedge \neg C)) \equiv (\neg A \vee B) \wedge (\neg A \vee \neg C) \wedge (\neg B \vee C \vee A)$

(j) True or False? If a first-order logical sentence α does not contain the constant (i.e. zero-argument function) A, then $\exists x(Bird(x) \wedge Child(x, Bob)) \models \alpha$ if and only if $(Bird(A) \wedge Child(A, Bob)) \models \alpha$.

2 Resolution Closure (5 pts)

What is the resolution closure of $\{A \vee B \vee \neg C, A \vee \neg B \vee C, \neg A \vee D\}$? Do not include clauses that are logically equivalent to True.

3 The Social Club (15 pts)

There's a social club that you want to join. As a condition of entry, they offer you two drinks, one of which is poisoned and one of which is not. You are told you must choose one to drink. For some reason, you still really want to join the club.

Conveniently, each member of the club is either a truthteller (truthtellers always tell the truth) or a liar (liars always lie). To help yourself make the decision, you need to create a first-order logic formulation of the situation.

Use the following signature:

- $\mathsf{Drink1}$ is a zero-argument function that represents the first drink
- $\mathsf{Drink2}$ is a zero-argument function that represents the second drink
- $\mathsf{Poisoned}(d)$ is a one-argument predicate that indicates whether d is the poisoned drink
- $\mathsf{Member}(m)$ is a one-argument predicate that indicates whether m is a club member
- $\mathsf{Liar}(m)$ is a one-argument predicate that indicates whether m is a liar
- $\mathsf{SayYes}(m, d)$ is a two-argument predicate that indicates whether m would say "yes, drink d is poisoned" if asked

Express the following in first order logic:

(a) "Exactly one drink is poisoned."

(b) "Exactly one club member is a liar."

(c) "When asked about Drink1, a liar always lies about whether Drink1 is poisoned and a truthteller always tells the truth about whether Drink1 is poisoned."

4 Resolution for 2SAT and 3SAT (10 pts)

In class, we used code for a resolution solver that looked something like this:

```
def resolution_solver(C):
    # C is a list of CNF clauses

    class LocalState:
        processed = set()
        unprocessed = C

    def process_next_clause():
        next_to_process = local.unprocessed[0]
        local.unprocessed = local.unprocessed[1:]
        for clause in local.processed:
            for resolvent in resolve(next_to_process, clause):
                if resolvent not in local.processed | set(local.unprocessed):
                    local.unprocessed.append(resolvent)
                if resolvent.is_false(): # Checks if resolvent is False
                    return False
        local.processed.add(next_to_process)
        return True

    local = LocalState()
    while len(local.unprocessed) > 0:
        if not process_next_clause():
            return False
    return True
```

Suppose that the signature contains n symbols, and that a call to resolve(c1, c2) returns the list of clauses that can be obtained by resolving clause c1 and clause c2. As usual, assume that a clause cannot contain both the positive and negative literal for the same symbol (e.g. $C \vee \neg C$ is not a clause). Also assume that two clauses that contain the exact same literals are the same clause (e.g. $A \vee \neg B$ should be considered to be the same clause as $\neg B \vee A$).

Consider a call to resolution_solver(C):

(a) Suppose every clause in C contains **at most 2 literals**. Give a (tight) upper bound, in terms of n, on the number of calls to process_next_clause. Justify your answer. You may use big-oh notation if preferred.

(b) Suppose every clause in C contains **at most 3 literals**. Give a (tight) upper bound, in terms of n, on the number of calls to process_next_clause. Justify your answer. You may use big-oh notation if preferred.