CS 543: Homework Assignment 5

Aditya Mehrotra
amehrotra@wpi.edu

1 Chp 7, Ex. 7.3

a We can assume the language has predefined functions for if, or, and, not, if and only if, etc.

---

function PL-True? (s,m) **returns** *true, or false*
  if s = *True* **then return** *true*
  else if s = *False* **then return** *false*
  else if SYMBOL?(s) **then return** LOOKUP(s, m)
  else branch on the operator of s
    $\neg$: **return not** PL-True?(ARG1(s), m)
    $\bigvee$: **return** PL-True?(ARG1(s),m) **or** (ARG2(s), m)
    $\bigwedge$: **return** PL-True?(ARG1(s), m) **and** (ARG2(s), m)
    $\Longrightarrow$: **not** PL-True?(ARG1(s), m) **or** (ARG2(s), m)
    $\Leftrightarrow$: **return** PL-True?(ARG1(s), m) **iff** (ARG2(s), m)

---

b In all such models, the sentences P $\bigwedge$ ¬P, True $\bigvee$ ¬P , and False $\bigwedge$ P may be true or false in a partial model.

c A general algorithm must be able to handle empty partial models without any assignments. In such cases, the algorithm has to determine unsatisfiability or/and validity, which are NP-complete and co-NP-complete.

d If and and or analyze their arguments in order, terminating on false or true arguments, it serves better. In such scenario, the algorithm already has the necessary properties: P $\bigvee$ Q returns true in the partial model when P is true and Q is unknown, and ¬P $\bigwedge$ Q returns false in the partial model where P is true and Q is unknown. The truth values of Q $\bigvee$ true, Q $\bigvee$ ¬ Q and Q $\bigwedge$ ¬ Q, on the other hand, are not discovered.

e The use of early termination in Boolean operators will result in a significant performance improvement. Because the Boolean operators in most languages already have the desired characteristic, you'd have to build special unintelligent versions and expect a lag.

2 Chp 7, Ex. 7.6

a Since this follows monotonicity, this statement is True.

b True. If $(\beta \bigwedge \gamma)$ is True for all models of , then β and γ are true for all models of α. So, this implies $\vDash \beta$ and $\vDash \gamma$.

c False. This statement doesn't hold for $\beta \equiv \alpha$; $\gamma \equiv \neg\alpha$

3 Chp 7, Ex. 7.18

a A statement is considered satisfiable if certain truth value assignments can make the statement true. A statement is considered valid if all truth value assignments to the variables make it true.

A simple truth table has eight rows and gets a true output for all cases, hence the statement is valid.

b Considering the LHS
(Food $\implies$ Party) $\vee$ (Drinks $\implies$ Party)
($\neg$ Food $\vee$ Party) $\vee$ ($\neg$ Drinks $\vee$ Party)
($\neg$ Food $\vee$ Party $\vee$ $\neg$ Drinks $\vee$ Party)
($\neg$ Food $\vee$ $\neg$ Drinks $\vee$ Party)
Now consider RHS,
(Food $\wedge$ Drinks) $\implies$ Party
$\neg$ (Food $\wedge$ Drinks) $\vee$ Party
($\neg$ Food $\vee$ $\neg$ Drinks) $\vee$ Party
($\neg$ Food $\vee$ $\neg$ Drinks $\vee$ Party)
Clearly, LHS = RHS. Since the statement is of the form S $\implies$ S, it is valid.

c If the negation of the statement is unsatisfiable, the statement must be valid.
$\neg$[[(Food $\implies$ Party) $\vee$ (Drinks $\implies$ Party)] $\implies$ [(Food $\wedge$ Drinks) $\implies$ Party]]
[(Food $\implies$ Party) $\vee$ (Drinks $\implies$ Party)] $\wedge$ $\neg$ [(Food $\wedge$ Drinks) $\implies$ Party]
($\neg$ Food $\vee$ $\neg$ Drinks $\vee$ Party) $\wedge$ Food $\wedge$ Drinks $\wedge$ $\neg$ Party
Each clause here is negated with itself, thus, resulting in an empty set.