National University of Singapore
School of Computing
CS3243: Foundations of Artificial Intelligence
Solutions for Tutorial 6

Readings: AIMA Chapter 7 & 8

1. The *WalkSAT* algorithm is a local search algorithm used to determine whether a proposition is entailed by a KB (whether the resulting KB is satisfiable). It is similar to simulated annealing in that it uses randomness and allows steps that generate more conflicts to be taken with some probability. The algorithm is shown below.

```
function WalkSAT(clauses, p, max-flips) returns a satisfying model or failure
   inputs: clauses, a set of clauses in propositional logic
           p, the probability of choosing to do a "random walk" move, typically around
0.5
           max-flips, number of flips allowed before giving up

   model ← a random assignment of true/false to the symbols in clauses
   for i = 1 to max-flips do
      if model satisfies clauses then return model
      clause ← a randomly selected clause from clauses that is false in model
      with probability p flip the value in model of a randomly selected symbol from
clause
      else flip whichever symbol in clause maximizes the number of satisfied clauses
   return failure
```

On the other hand, `DPLL` is a deterministic model checking algorithm. It uses pure symbols and unit clauses as the basis for heuristics that attempt to converge to a solution quicker than the standard `TT-Entails`.

(a) How you would modify the WalkSAT algorithm to use the heuristics of pure symbols and unit clauses.

Given the current assignment of truth values to literals, WalkSAT can look for pure symbols and unit clauses and probabilistically favor flipping truth values of these literals to the truth value that would minimize conflicts.

To implement this efficiently, each time a literal is chosen and its truth value flipped, a update procedure must be run to calculate the resulting pure symbols and unit clauses, along with number of conflicts. We might also prefer to pick literals to flip that generate a larger number of pure symbols and unit clauses (along with minimizing conflicts) such that future steps may have a larger impact.

(b) How would such a modification affect the performance of the resulting algorithm? How does it impact time complexity?

Such a modification does not change the asymptotic complexity of the algorithm, as the worst case running time is the same. However, it does complexity to the algorithm, as now the checks for unit clauses and pure symbols need to be done in conjunction with the calculation of the number of conflicts.

2. (Question 8.2 from AIMA) Consider a knowledge base containing just two sentences: $P(a)$ and $P(b)$. Does this knowledge base entail $\forall x \; P(x)$? Explain your answer in terms of models.

   No. Consider the model with domain $\{O_1, O_2, O_3\}$ *(i.e., 3 objects), and the interpretation where the constant symbol* a *refers to* O_1 *and the constant symbol* b *refers to* O_2*, and the predicate symbol* P *refers to the relation* $\{< O_1 >, < O_2 >\}$*. Then* $P(a)$ *and* $P(b)$ *are true but* $\forall x \; P(x)$ *is false.*

3. (Question 8.3 from AIMA) Is the sentence $\exists x, y \; x = y$ valid? Explain.

   Yes. Consider any model with domain D*. Since* D *is always nonempty, let* O *be some object in* D*. Then the interpretation that assigns both* x *to* O *and* y *to* O *is such that* $x = y$ *is true. Hence* $\exists x, y \; x = y$ *is true in every model and is valid.*

4. (Wumpus World) Represent the following English sentences in first-order logic:

   (a) Anyone who meets the wumpus is killed by it.
   $\forall x \; Meets(x, Wumpus) \Rightarrow Kills(Wumpus, x)$

   (b) Anything that glitters is gold.
   $\forall x \; Glitters(x) \Rightarrow Gold(x)$

   (c) Not every square contains a pit.
   $\neg \forall x \; Square(x) \Rightarrow Contains(x, Pit)$ or
   $\exists x \; Square(x) \wedge \neg Contains(x, Pit)$

5. (Modified Question 8.6 from AIMA) Represent the following sentences in first-order logic, using a consistent vocabulary that you must define:

   We define the following vocabulary:

   $took(x, y, z)$ *: is true student x took subject y in semester z*
   $score(x, y, z)$ *: is true if student x obtains score z in subject y*
   $passed(x, y)$ *: is true if student x passed subject y*
   $buys(x, p)$ *: is true if person x buys policy p*
   $isSmart(x)$ *: is true if person x is smart*
   $isExpensive(x)$ *: is true if x is expensive*
   $sells(x, y, p)$*: is true if person x sells policy p to person y*
   $isInsured(x)$*: is true if person x is insured*
   $isBarber(x)$*: is true if x is a barber*
   $shaves(x, y)$*: is true if person x shaves person y*

   (a) Some students took French in Spring 2001.
   $\exists x \; took(x, French, Spring2001)$

   (b) Every student who takes French passes it.
   $\forall x, y \; took(x, French, y) \Rightarrow passed(x, French)$

   (c) Only one student took Greek in Spring 2001.
   $\exists x, \; took(x, Greek, Spring2001) \wedge$
   $\forall (y, z \; took(y, Greek, Spring2001) \wedge took(z, Greek, Spring2001) \Rightarrow y = z)$

   (d) The best score in Greek is always higher than the best score in French.
   $\exists x, s \, \forall y, t \; score(x, Greek, s) \wedge score(y, French, t) \wedge s > t$

(e) Everyone who buys a policy is smart.
$\forall x, p \; buys(x, p) \Rightarrow isSmart(x)$

(f) No person buys an expensive policy.
$\neg\exists x, p \; buys(x, p) \wedge isExpensive(p)$

(g) There is an agent who sells policies only to those people who are not insured.
$\exists x \forall y, p \; sells(x, y, p) \Rightarrow \neg isInsured(y)$

(h) There is a barber who shaves all men in town who do not shave himself.
$\exists x \forall y \; isBarber(x) \wedge \neg shaves(y, y) \Leftrightarrow shaves(x, y)$

6. (Modified Question 8.7 from AIMA) Represent the sentence "All Germans speak the same languages" in predicate calculus. Use $Speaks(x, l)$ to specify that a person x speaks language l and $isGerman(x)$ to specify that a person x is a German.

$$\forall x, y, l \; isGerman(x) \wedge isGerman(y) \wedge Speaks(x, l) \Rightarrow Speaks(y, l)$$