Course: CSC520, Introduction to Artificial Intelligence
Homework 1
Student: Xusheng Xiao

---

1. **(12 points) Describe PEAS for the following:**

   (a) Bot to display advertisements in a search engine (eg. Bing, Google etc.)

   (b) Industrial robot (eg. detect surface defects on automobile body in assembly line)

   (c) Recommendation system (eg. Amazon book suggestion system)

   In each case, state whether the environment is fully observable, deterministic, episodic, and single agent.

Agent Type	**Performance Measure**	**Environment**	**Actuators**	**Sensors**
AD Bot in search engines	Highly relevant to search condition, low prize, free shipping	users, keyboards, screen	Display ADs	Keyboard entry of search condition and search history
Industrial robot	Maximize precision of detecting defects, low cost	Product parts, assemble line	Display detected defects	Scanned image data
Recommendation system	Recommended books are highly related to user preference and most possible for users to purchase	Users, recommendation system	Display recommended books	Keyboard entry of search words, history of purchased book

Task Environment	**Observable**	**Deterministic**	**Episodic**	**Agents**
AD Bot in search engines	Partially	Stochastic	Sequential	single agent
Industrial robot	Fully	Deterministic	Episodic	single agent
Recommendation system	Partially	Stochastic	Sequential	single agent

2. **(18 points) Answer the following questions from the textbook: 2.6a, 2.6b, 2.12**

(a) This exercise explores the difference between agent functions and agent programs.

i. Can there be more than one agent program that implements a given agent function? Given an example, or show why one is not possible.

ii. Are there agent functions that cannot be implemented by any agent program?

Answer:

i. There can be multiple programs that implement a given agent function. For example, for the vacuum agent problem shown in textbook, using table to map the percept to action is one implementation. However, we can also compute the hash code of the percept by adopting some hash function, and then use the computed hash code to look up for the corresponding action. This hash-look-up is another implementation for the same vacuum agent problem. Thus, there can be multiple programs that implement a given agent function.

ii. I believe the answer is "there is not any function cannot be implemented by any agent program". The scientific research is evolving fast in modern society. A few years ago, touch screen only can support very basic functionalities for human to communicate with computers. Using just several fingers to do zooming and rotating seems like an agent function that cannot be implemented. Nowadays, with the advances of multi-touch technologies, users now can touch the screen with multiple fingers and using different combination of fingers to send different signals to PC for different actions, such as zooming and rotating objects.

(b) Consider a modified version of Exercise 2.8, in which the geography of the environment - its extent, boundaries, and obstacles - is unknown, as is the initial dirt configuration. (The agent can go *Up* and *Down* as well as *Left* and *Right*.) Repeat this exercise for the case in which the location sensor is replaced with a "bump" sensor that detects the agent's attempts to move into obstacles or to cross the boundaries of the environment. Suppose the bump sensor stops working; how should the agent behave?

i. Can a simple reflex agent be perfectly rational for this environment? Explain.

Since the bump sensor stops working, which makes the agent cannot get the correct information about obstacles, extent and boundaries, a simple reflex agent cannot work perfectly. For example, if we define a rule that if there is no dirt on the current location, move right. In the situation where the right location has an obstacle and moving right cannot achieve, the simple reflex agent cannot have the other reaction and only can stay in the same location forever.

ii. Can a simple reflex agent with a *randomized* agent function outperform a simple reflex agent? Design such an agent and measure its performance on several environments.
With the bump sensor not working, a randomized agent should perform better than a simple reflex agent. The reason is that with the simple reflex agent, there must be a correct percept of the environment for it to do the right action. But with the bump sensor not working, a simple reflex agent cannot react correctly. However, for a randomized agent, it would pick random action based on the given percept. Thus, it is possible that it can correctly pick up the right action with the bump sensor not working. The design could be quite simple: The vacuum sucks up when it detects the current location has dirt; every time when the agent detects that there is no dirt on the current location, it randomly picks up a possible direction and move.

iii. Can you design an environment in which your randomized agent will perform poorly? Show your results.
To make a randomized agent perform poorly, we can design an environment that requires the agent to move several times of the same direction, say *Up*. Since it is randomized, the probability for it to choose *Up* action continuously for several times is very low. It is very possible that it moves up and then the other direction, which makes it difficult to move "up" for several times.

iv. Can a reflex agent with state outperform a simple reflex agent? Design such an agent and measure its performance on several environments. Can you design a rational agent of this type?
A state reflex agent can outperform a simple reflex agent. We can design it like this: If there is no dirt on the current location, move right and save the direction in a state. If last time's moving can not succeed, change the direction clock-wisely, and try to move again. As a result, such a state reflex agent can find a way around obstacles and reach a location where there is dirt.

3. **(20 points) Introducing our agent Mr.Wuf who will help us moving things from one place to another. One day, Mr.Wuf is assigned a task of transferring a set of boxes one-by-one by lifting them from location A and placing them in location B inside a building. A signalling system says whether the agent is near its destination or not. The room has stationary obstacles whose locations are unknown. If the agent bumps into an obstacle, the box in hand will fall down and some boxes have fragile goods. But there are safe paths, some longer than the others. Your must help Mr.Wuf with the task by answering the following questions. Mr.Wuf does not have enough time !!!**

   (a) Define PEAS.

(b) Is it sufficient for Mr.Wuf to be simple reflex ? Why or why not ?

(c) Mr.Wuf likes to move randomly. To what extent would this help ? Are there drawbacks ?

(d) Suggest one improvement to Mr.Wuf's design. Does your improvement have drawbacks ?

Answer:

(a)

Agent Type	Performance Measure	Environment	Actuators	Sensors
Mr. Wuf	Find a shortest safe path for moving all the boxes from Location A to B in the building	Location A & B in a building, obstacles, boxes	hands for lifting boxes and legs for walking	Signalling system, eyes

(b) It is not sufficient. When the signalling system says that it is still not the destination, Mr. Wuf should move. Since Mr. Wuf is simple reflex, he can only choose one direction to move. If by moving in that direction he would face an obstacle, then he will still do that since he is simple reflex and can only move that direction by receiving the percept saying that current location is not destination. After he falls, he goes back to Location A and moves another box. However, since he is simple reflex, he would choose the same path as last time and fall again. As a result, he would fail the mission.

(c) Moving randomly can help Mr. Wuf a little. For example, when there is an obstacle on the right of Mr. Wuf, he may randomly choose to move up and then right to avoid that.

(d) Mr. Wuf can perform better if he is simple reflex with state. Whenever Mr. Wuf reaches a new location, save the location and the direction in a state. If he moves right on one location and then bump into an obstacle, he also save it. Later when he reaches the same location, he can change the direction clock-wisely and move. The drawback of it is it may not find an optimal path.

4. **(30 points) Consider the following english sentences.**

(a) Marcus was a man

(b) Marcus was a Pompeian

(c) All Pompeians were Romans

(d) Caesar was a ruler

(e) All Romans were either loyal to Caesar or hated him

(f) Everyone is loyal to someone

(g) Any man only tries to assassinate rulers he is not loyal to

(h) Marcus tried to assassinate Caesar

Now answer the following questions:

(a) Convert the above into first order predicate logic using the following predicates: ruler, man, Pompeian, Roman, loyalto, hate, tryassassinate Use appropriate quantifiers and connectors.

(b) Convert the above sentence to CNF.

(c) Answer the following questions using resolution discussed in class based on the knowledge above:

i. Was Marcus loyal to Caesar ?

ii. Did Marcus hate Caesar ?

Answer:

(a) FOPL:

i. $man(Marcus)$

ii. $Pompeian(Marcus)$

iii. $\forall X(Pompeian(X) \Rightarrow Roman(X))$

iv. $ruler(Caesar)$

v. $\forall X(Roman(X) \Rightarrow loyalto(X, Caesar) \vee hate(X, Caesar))$

vi. $\forall X(\exists Y(loyal(X, Y)))$

vii. $\forall X(\forall Y((man(X) \wedge ruler(Y) \wedge tryassassinate(X, Y)) \Rightarrow \neg loyalto(X, Y)))$

viii. $tryassassinate(Marcus, Caesar)$

(b) CNF:

i. $man(Marcus)$

ii. $Pompeian(Marcus)$

iii. $\neg Pompeian(X1) \vee Roman(X1)$

iv. $ruler(Caesar)$

v. $\neg Roman(X2) \vee loyalto(X2, Caesar) \vee hate(X2, Caesar)$

vi. $loyal(X3, a)$

vii. $\neg man(X4) \vee \neg ruler(Y1) \vee \neg tryassassinate(X4, Y1) \vee \neg loyalto(X4, Y1)$

viii. $tryassassinate(Marcus, Caesar)$

(c) **Answers:**

i. Conclusion: $\neg loyalto(Marcus, Caesar)$

Negated conclusion:
1. $loyalto(Marcus, Caesar)$

2.	$\neg ruler(Y1) \lor \neg tryassassinate(Marcus, Y1) \lor \neg loyalto(Marcus, Y1)$	$i + vii\{Marcus/X4\}$
3.	$\neg tryassassinate(Marcus, Caesar) \lor \neg loyalto(Marcus, Caesar)$	$iv + 2\{Caesar/Y1\}$
4.	$\neg loyalto(Marcus, Caesar)$	$viii + 3\{\}$
5.	$\emptyset$	$4 + 1\{\}$

ii. Conclusion: $hate(Marcus, Caesar)$

Negated conclusion:
1. $\neg hate(Marcus, Caesar)$

2.	$\neg ruler(Y1) \lor \neg tryassassinate(Marcus, Y1) \lor \neg loyalto(Marcus, Y1)$	$i + vii\{Marcus/X4\}$
3.	$\neg tryassassinate(Marcus, Caesar) \lor \neg loyalto(Marcus, Caesar)$	$iv + 2\{Caesar/Y1\}$
4.	$\neg loyalto(Marcus, Caesar)$	$viii + 3\{\}$
5.	$Roman(Marcus)$	$ii + iii\{Marcus/X1\}$
6.	$loyalto(Marcus, Caesar) \lor hate(Marcus, Caesar)$	$5 + v\{Marcus/X2\}$
7.	$hate(Marcus, Caesar)$	$6 + 4\{\}$
8.	$\emptyset$	$7 + 1\{\}$

5. **(20 points) Consider the following English statements:**

   (a) John is a graduate student

   (b) Graduate students buy cheaper books

   (c) AI book is costly

   (d) The neighborhood store "Bookmarks" has a discount on the AI book

   (e) Books on discount are cheap

   Now, using the resolution approach for first order predicate logic discussed in class, answer: "Will John buy the AI book from BookMarks?"

   Lexicon:
   Constant: John, AI book, Bookmarks
   Variable: X,Y
   Predicate:

$graduate(X)$	– X is a graduate student
$book(X)$	– X is a book
$cheap(X)$	– X is cheap
$buy(X, Y)$	– X buys Y
$store(X)$	– X is a store
$discount(X)$	– X has discount

   FOPL:

   (a) $graduate(John)$

(b) $\forall X, Y(graduate(X) \wedge book(Y) \wedge cheap(Y) \Rightarrow buy(X, Y))$

(c) $book(AIbook) \wedge costly(AIbook)$

(d) $store(Bookmarks) \wedge discount(AIbook)$

(e) $\forall X(book(X) \wedge discount(X) \Rightarrow cheap(X))$

CNF:

(a) $graduate(John)$

(b) $\neg graduate(X1) \vee \neg book(Y1) \vee \neg cheap(Y1) \vee buy(X1, Y1)$

(c) $book(AIbook)$

(d) $costly(AIbook)$

(e) $store(Bookmarks)$

(f) $discount(AIbook)$

(g) $\neg book(X2) \vee \neg discount(X2) \vee cheap(X2)$

Conclusion: $buy(John, AIbook)$ – John buy the AI book from BookMarks.

Negated conclusion:
1. $\neg buy(John, AIbook)$

2.	$\neg discount(AIBook) \vee cheap(AIBook)$	$g + c\{AIBook/X2\}$
3.	$cheap(AIBook)$	$f + 2\{\}$
4.	$\neg book(Y1) \vee \neg cheap(Y1) \vee buy(John, Y1)$	$b + a\{John/X1\}$
5.	$\neg cheap(AIBook) \vee buy(John, AIBook)$	$4 + c\{AIBook/Y1\}$
6.	$buy(John, AIBook)$	$5 + 3\{\}$
7.	$\emptyset$	$1 + 6\{\}$