1. Definitions of Task Environment
   1. Environment - Where the agent acts
      1. Online, against a possible database of your transactions
   2. Actuators - how the agent will perform its actions
      1. Check your habits vs a range of accepted habits.
   3. Sensors - how the agent will perceive the environment
      1. Data from database
2. Fraud detection performance measures
   1. For performance measures, we will want to get some form of feedback for the AI. both positive and negative feedback. In this case, Positive things can involve things like:
      1. Confirmed fraud
   2. Negative modifiers could be:
      1. A false positive
      2. Time taken
3. Describe environments according to following properties:
   1. Fully vs partially
      1. From what I understand, the AI for this program can only fall under partially observable, it can make inferences on when your information is used, but it cannot precisely say that you were the one to use it or not
   2. Deterministic vs stochastic
      1. I believe this section to be stochastic, we have a partially observable environment, because we are working within a partially observable environment.
   3. Episodic vs sequential
      1. This appears to be Episodic, where the agent receives the information, makes a comparison, and is done. A decision now does not depend on a previous decision.
   4. Static vs dynamic
      1. I think this one is potentially dynamic, where the agent does not have all the time in the world to make the decision, another purchase may be made by the individual who has stolen the information, so the agent needs to be able to see this.
   5. Discrete vs continuous
      1. According to the slides, this should be continuous. As it expects values that have an amount of variance, but they should still fall within a range.
   6. Single vs multi-agent
      1. As noted in the given problem, only one agent is listed, so this should be single agent.
4. I believe that the choice should be a model-based reflex agent. We can learn some of the habits that the user has within the first few weeks, and use this to gain some insight into how they will behave in the future. The agent will compare the habits of of the account owner, and it notices strange patterns, then it will have a reflex of freezing assets until the owner confirms.
5. Vacuum cleaner
   1. I do not believe that a simple reflex agent would be a good solution for the vacuum cleaner, while we have a map of the room, we are not aware of what spaces have dirt, and is also unaware if it has been at that position previously. Therefore, this agent would just run into an infinite loop.
   2. Reflex Agent

The Reflex agent should return an action based on a set of rules and conditions

Percepts: Locations, and contents (when on square)

Actions: left, right, suck, do nothing

Function Simple\_Reflex\_Agent (percept) returns an action

Static: rules, a set of conditions\_actions rules

State <- Interpret\_Input (percept)

Rule <- Rule\_Match (state rules)

Action <- Rule\_Action [rule]

Return action

Function Reflex\_Vacuum\_Agent([location status]) returns an action

If status == dirty then return SUCK

Else if location = A then return RIGHT

Else if location = B.Max return Do Nothing

Else if location = B then return LEFT

//\*\*\*\*\*\*

If at max B, then we want to stop, if we did not, we would encounter an infinite loop

//\*\*\*\*\*\*

* 1. If given the case, a simple reflex agent could be rational, because the environment would then be fully observable. The agent could use a table to compare positions and when they have been updated, then the precept will tell it what to do next. Something like this:

|  |  |
| --- | --- |
| Percept Sequence | Action |
| [A, Clean] | Right |
| [A, Dirty] | Suck |
| [B, Dirty] | Suck |
| [B, Clean] | Left |
| [A, Clean][A, Clean] | Right |
| [A, Clean][A, Dirty] | Suck |