Homework #1: Chapters 1–3.4 Solutions

1. [15 points] Develop a PEAS description for the following task environments:

   a) A flying package delivery drone that delivers small packages to people’s houses.
      ■ P: minimize delivery time, (if quantified) protect package, protect itself
      ■ E: static obstacles (e.g. buildings, trees), aerial obstacles (e.g. birds, other drones, people), map (with addresses), weather conditions
      ■ A: propellers, package grabbing arm, siren (to avoid collisions)
      ■ S: camera, barometer, GPS, gyroscopes

   b) A software agent that can play a computerized version of Freecell
      ■ P: maximize number of cards moved to foundation, minimize total number of moves, minimize time taken
      ■ E: the various card piles (foundations, cascades, free cells)
      ■ A: can move individual cards
      ■ S: the position of each of the cards, a timer

   c) An agent that attempts to acquire an item via an online auction site such as eBay
      ■ P: minimize cost of the item, minimize time taken to acquire, avoid acquiring extra items (e.g. bundles, or the agent is bidding in multiple auctions simultaneously)
      ■ E: the Internet, or possibly just the eBay site itself
      ■ A: can place bids on auctions
      ■ S: web scraper and parser

2. [15 points] For each of the agents described above, categorize it with respect to the six dimensions of task environments

   a) A flying package delivery drone that delivers small packages to people’s houses.
      ■ Partially observable. The agent is only aware of obstacles within the limited range of its sensors.
      ■ Multi-agent. Obstacles (e.g. birds) may interfere with the agent
      ■ Stochastic. The real world is not predictable.
      ■ Sequential. If the drone crashes into something, it may impair its ability to fly well later on
      ■ Dynamic. The world changes regardless of the drone’s actions
      ■ Continuous. The real world does not move in discrete steps.

   b) A software agent that can play a computerized version of Freecell
      ■ Fully observable. Every card is face-up
      ■ Single agent. Freecell is a solitaire game
      ■ Deterministic. Card movements always succeed or always fail
      ■ Sequential. An earlier bad move can make the hand unwinnable
      ■ Static. The cards do not move unless the agent moves them
      ■ Discrete. Each card has a finite number of positions it can be in.
c) An agent that attempts to acquire an item via on an online auction site such as eBay
   ■ Fully observable in the sense that eBay is a public website. Partially observable in the
   sense that the whole site may be too large for the agent to search/scrape and there is
   hidden information regarding the other bidders on items
   ■ Multi agent competitive. The other bidders are agents competing to win the item
   ■ Deterministic. The rules of eBay do not involve randomness
   ■ Sequential. Previous bids may change the behavior of bidders e.g. a bidding war
   ■ Dynamic. Other bidders will continue to act and auctions timers will count down even
   if the agent does nothing
   ■ Continuous in the sense that bids are time-sensitive (time is continuous). Discrete in
   the sense that everything on a computer (such as eBay) is ultimately discrete.

3. [15 points] Consider the vacuum-cleaner world depicted in Figure 2.2, where the agent
   perceives which square it is in and whether there is dirt in the square. Consider the following
   three questions assuming that the agent’s performance measure is reduced each time it moves
   left or right. (New dirt may appear, if the student specifies)
   a) A simple reflex cannot be rational. It only sees the current percept, so it does not know if
      the other square is clean. Thus it will either a) clean one square and then stop (possibly
      leaving a dirty square) or b) clean both squares but keep moving back and forth forever
      (driving down the performance measure). Neither behavior is rational. Even if dirt can
      reappear, it is not rational to keep moving back and forth unless dirt is likely to ever other
      move. A rational agent would wait a fixed time before checking the other square, in order
      to minimize unneeded moves.
   b) A model-based reflex agent can be rational. It can clean a square, clean the other square,
      and then stop (remembering that it already cleaned the previous square). If dirt can
      reappear, it can wait in a given square for a fixed time before moving to check the other
      square.
   c) If the agent’s percepts give it the clean/dirty status of every square in the environment at
      each time step, then the simple reflex agent (as in Figure 2.8) can be rational as well. It
      will receive a percept saying both squares are clean, in which case it stops. Or in the case
      that dirt can reappear, it only moves when its square is clean and the other square is dirty.

4. [25 points] Using the road map of Romania from the book (Figure 3.2), find a path from
   Pitesti to Sibiu using breadth-first graph-search. Assume that when all else is equal, cities are
   chosen in alphabetical order.
Breadth-first search is optimal in this case, but in general, there is no guarantee that it will be because the step costs are not equal. Consider for example if there were four towns along the way between Riminciu Vilcea and Sibu. Then the roundabout path through Bucharest and Fargaras would have been found first.

5. **[10 points]** A finite state space always has a finite search tree if it has no cycles or if the search algorithm has a method of detecting and avoiding cycles. Consider the vacuum-world agent moving back and forth forever, for example.

6. **[20 points]** You are standing on a river bank with a wolf, a goat, some cabbage, and a boat. You want to get the wolf, goat, and cabbage across the river however the boat can only hold yourself and one other item. Additionally, the wolf will eat the goat if left alone and the goat will eat the cabbage if left alone.

   We can represent this state space with a factored representation with four factors: you, wolf, goat, cabbage. Each factor can have the value left or right (indicating which bank it is on) or eaten. For short we represent each state with four letters, where each letter is L, R, or E.

   Our initial state is LLLL and our goal test checks for the state RRRR.

   Possible actions are: take wolf, take goat, take cabbage, or take nothing. To find a solution that minimizes the number of trips, we make each action cost 1.

   Our transition model indicates when these actions are possible and what their effects are:

   - **Take wolf:** applies if the first two factors are the same. Result is the first two factors switch from L to R or vice versa. Also if the last two factors were the same as the first two, the last factor switches to E. For example LLRL→RRRL or LLLL→RRLE. Alternatively, the action is not available if the last two factors are the same.
   
   - **Take goat:** applies if the first and third factors are the same. Result is the first and third factors switch from L to R or vice versa. For example LRLR→RRRR
   
   - **Take cabbage:** applies if the first and last factors are the same. Result is the first and last factors switch from L to R or vice versa. Also if the second and third factors were also the same as the first two, the third factors switches to E. For example RRLR→LRLL or LLLL→RLER
   
   - **Take nothing:** always applies. Result is the first factor switches from L to R or vice versa. Also if the second and third factor were the same as the first, the third switches to E. And if the third and fourth factor were the same as the first, the fourth switches to E. For example LRLR→RRLR or LLLL→RLEE.