Sheet 2

1. (2.8) Implement a performance-measuring environment simulator for the vacuum-cleaner world depicted in Figure 2.2 and specified on page 38. Your implementation should be modular so that the sensors, actuators, and environment characteristics (size, shape, dirt placement, etc.) can be changed easily. (Note: for some choices of programming language and operating system there are already implementations in the online code repository.)

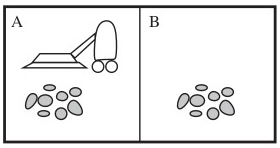

Figure 2.2 A vacuum-cleaner world with just two locations.

Consider the simple vacuum-cleaner agent that cleans a square if it is dirty and moves to the other square if not; this is the agent function tabulated in Figure 2.3. Is this a rational agent? That depends! First, we need to say what the performance measure is, what is known about the environment, and what sensors and actuators the agent has. Let us assume the following:

- The performance measure awards one point for each clean square at each time step, over a "lifetime" of 1000 time steps.
- The "geography" of the environment is known *a priori* (Figure 2.2) but the dirt distribution and the initial location of the agent are not. Clean squares stay clean and sucking cleans the current square. The *Left* and *Right* actions move the agent left and right except when this would take the agent outside the environment, in which case the agent remains where it is.
- The only available actions are *Left*, *Right*, and *Suck*.
- The agent correctly perceives its location and whether that location contains dirt.

We claim that *under these circumstances* the agent is indeed rational; its expected performance is at least as high as any other agent's. Exercise 2.2 asks you to prove this.

2. (2.9) Implement a simple reflex agent for the vacuum environment in Exercise 2.8. Run the environment with this agent for all possible initial dirt configurations and agent locations. Record the performance score for each configuration and the overall average score.
3. (2.10) Consider a modified version of the vacuum environment in Exercise 2.8, in which the agent is penalized one point for each movement.
   (a) Can a simple reflex agent be perfectly rational for this environment? Explain.
   (b) What about a reflex agent with state? Design such an agent.
   (c) How do your answers to a and b change if the agent's percepts give it the clean/dirty status of every square in the environment?
4. (2.11) Consider a modified version of the vacuum environment in 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.)
   (a) Can a simple reflex agent be perfectly rational for this environment? Explain.
   (b) 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.
   (c) Can you design an environment in which your randomized agent will perform poorly? Show your results.
   (d) 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?
5. (2.12) Repeat Exercise 2.11 for the case in which the location sensor is replaced with a "bump" sensor that detects the agent's attempts to move into an obstacle or to cross the boundaries of the environment. Suppose the bump sensor stops working; how should the agent behave?
6. (2.13) The vacuum environments in the preceding exercises have all been deterministic. Discuss possible agent programs for each of the following stochastic versions:
   (a) Murphy's law: twenty-five percent of the time, the Suck action fails to clean the floor if it is dirty and deposits dirt onto the floor if the floor is clean. How is your agent program affected if the dirt sensor gives the wrong answer 10% of the time?
   (b) Small children: At each time step, each clean square has a 10% chance of becoming dirty. Can you come up with a rational agent design for this case?