

Program 2 30 points

Mars Explorer

Purely Reactive Agents

Background

The provided code attempts to solve the following problem:

The objective is to explore a distant planet. You are to collect samples of a particular type of precious rock. The location of the rock samples is not known in advance, but they are typically clustered in certain spots. A number of autonomous robots are available that can move around and collect samples and later re-enter the spacecraft to go back to earth. Each robot has a limit to how much sample it can hold at a given time. The location of the spaceship will be referred to as “the base”. There is no detailed map of the planet available, although it is known that the planet is full of obstacles (hills, valleys, etc) which will prevent the robots from exchanging information and may prevent them from traveling in certain areas.

In order to complete the task, there are two mechanisms which were developed by Steels (who first described the classic problem):

Basic Behavior

1. A *gradient field* emanating from the spaceship (and weakening the further away from home base one travels) helps the vehicles know how to return and how far away home base is. (We implement this feature merely by knowing the position of the spaceship in the grid.)
2. Indirect communication is facilitated by *sample crumbs* which can be dropped, picked up, and detected by passing robots.

The behavior of the robots is reactive:

- a. If you detect an obstacle, change directions.
- b. If you are carrying samples and are at the base, drop the samples.
- c. If you have a full load of samples and are not at the base, move towards the base (dropping crumbs along the way).
- d. If you detect a sample, pick up as much as you can.
- e. If you detect a sample in a neighboring cell, move to that cell.
- f. If you have nothing better to do, move randomly.

The code

Compile and run the sample code that is provided (Mars.jar).

The writer of the sample code got a little carried away with objects, but generally did a good job.

Take some time to familiarize yourself with the code. Note that each triangular agent is labeled by a pair of numbers (id:amount). Id is an identification number. Amount is the amount of sample it is carrying. Note, agents don't return to home base with small amounts of sample, but keep seeking a full load.

Home base is represented by a cyan-colored cell (at 0,0). It is labeled with the total amount of samples in the grid, the total steps taken, and (in red) the amount of sample currently at home base.

Your assignment

Study the existing code. Note, there are three sample boards (board0.dat, board1.dat, board2.dat).

Make the following modifications:

1. The agents now just wander around picking up and dropping crumbs. Implement the "basic behavior" described above. Stop the simulation after X steps (where you determine the value of X). The *performance* is determined by the total samples returned to home base minus the work. The work is the number of steps taken (cells entered) by all agents plus the number of steps it would take the agents to return to the space ship (assuming no obstacles). You will want to modify the GUI so it shows current performance.
2. Have the spaceship be able to "call" the robots back to the spaceship (under conditions you determine). When all robots return to the spaceship, the *performance* is determined by the total samples returned minus the number of steps taken (cells entered) by all agents.
3. Allow the robots to mark the cells with their "scent". The more times they have visited the cell, the stronger is the scent. Identify and implement a way of using this state information to improve the performance of the robots. Be sure to keep the flavor of a reactive agent. Feel free to create a new status for the agent (see AgentAction.java).
4. Create a "report.doc" file in which you include the following information:

Concept: (describe the concept behind your design)

Results: (Report the results of your improvements on each of three boards with varying number of agents.)

Board0.dat	Basic Code	Basic Code	Basic Code	Improved Code	Improved Code	Improved Code
Number	Amount	Number	Performance	Amount	Number	Performance

Of agents	of Sample	of Time Steps		of Sample	of Time Steps	
2						
3						

Board1.dat	Basic Code	Basic Code	Basic Code	Improved Code	Improved Code	Improved Code
Number Of agents	Amount of Sample	Number of Time Steps	Performance	Amount of Sample	Number of Time Steps	Performance
2						
3						

Board2.dat	Basic Code	Basic Code	Basic Code	Improved Code	Improved Code	Improved Code
Number Of agents	Amount of Sample	Number of Time Steps	Performance	Amount of Sample	Number of Time Steps	Performance
2						
3						