

11.3 Movement of Ants

Mathematica Quick Review Questions

Introduction to Computational Science: Modeling and Simulation for the Sciences

Angela B. Shiflet and George W. Shiflet

Wofford College

© 2006 by Princeton University Press

This file contains system-dependent Quick Review Questions and answers in *Mathematica* for Module 11.3 on "Movement of Ants." Complete all code development in *Mathematica*.

Grid Initialization

Quick Review Question 1 This question refers to the initialization of the grid for ant movement. Suppose we establish constants *EMPTY*, *NORTH*, *EAST*, *SOUTH*, and *WEST* as representative characters, as follows:

```
EMPTY = "T"; NORTH = "N"; EAST = "E"; SOUTH = "S"; WEST = "W";
```

- Write a statement to return a random integer between 1 and 4.
- We define a list, *dir*, of directions, as follows:

```
dir = {NORTH, EAST, SOUTH, WEST};
```

Suppose *r* is an integer between 1 and 4. Give the expression to return the direction from *dir* with index *r*.

- Complete the code to assign to *grid* an *n*-by-*n* table of ordered pairs. The first member of each ordered pair is zero. With a probability of *probAnt*, the cell contains an ant that faces in a random direction. Otherwise, the cell does not contain an ant.

```
grid =  
  _____ [ { _____ ,  
                If [ _____ < _____ ,  
                    dir [ [ Random [ Integer , { 1 , 4 } ] ] ] , _____ ] } ,  
  { n } , { n } ]
```

- Complete the loop to generate a chemical trail with no ants on the middle row of the grid *grid*. The maximum amount of chemical, 50, occurs in column *n*, and for column *j* the amount of chemical is a fraction, j/n , expressed as an integer, of the maximum 50. For example, if *j* is 10 and *n* is 17, then the amount of chemical is the integer 29 because $(50)(10)/17 = 29.41$.

```
_____ [
```

```
grid[[IntegerPart[n/2] + 1, _____]] = {_____, _____},
{j, 1, n}]
```

Sensing

Quick Review Question 2 With parameter a representing the amount of chemical, complete the code for the first *sense* rule that an empty cell does not sense.

```
sense[_____, __, __, __, __] := _____
```

Quick Review Question 3 This question refers to the second *sense* rule that an ant turns at random in the direction of one of the neighboring cells with the greatest amount of chemical. Assume the rule begins as follows:

```
sense[{a_, d_}, {n_, _}, {e_, _}, {s_, _}, {w_, _}] := ...
```

- a. Complete the assignment to *lst* of a list of the amounts of chemical in neighboring cells.

```
lst = _____
```

- b. Complete the statement to assign the maximum level from *lst* to *mx*.

```
mx = _____
```

- c. Complete the statement to assign to *posList* a list of indices in *lst* where the maximum level, *mx*, occurs.

```
posList = _____[_____ [_____, _____]]
```

- d. Complete the statement to assign to *lng* the number of elements in *posList*.

```
lng = _____[posList]
```

- e. Complete the statement to assign to *rndPos* a random integer between 1 and *lng*, inclusively, that is a possible index of *posList*.

```
rndPos = _____
```

- f. Complete the statement to assign to *dirIndex* the value from *posList* with index *rndPos*.

```
dirIndex= posList_____
```

- g. In Quick Review Question 1 we defined *dir* as follows:

```
dir = {NORTH, EAST, SOUTH, WEST}
```

With a being a parameter for the chemical level, give the ordered pair indicating chemical level and random direction in which to turn. Parts b-f developed the code to calculate the index of that direction, *dirIndex*.

- h. Give this entire *sense* rule.

Walking without Concern for Collision

Quick Review Question 4 Complete the definition of the first *walk* rule: For a cell that remains empty, the amount of chemical decrements by one but does not fall below 0.

`walk[{a_, EMPTY}, _, _, _, _, _, _, _, _, _, _, _, _] := _____`

Quick Review Question 5 Complete the definition of one form of the second *walk* bullet: If an ant wants to go north and the cell to the north is empty, the ant leaves the current cell and the amount of chemical at that site increments by one. Similar rules apply to the east, south, and west directions.

`walk[{a_, _____}, {_, _____}, _, _, _, _, _, _, _, _, _, _, _] := _____`

Quick Review Question 6 Complete the definition of the third *walk* rule: If an ant stays in a cell, the amount of chemical remains the same.

`walk[{a_, b_}, _, _, _, _, _, _, _, _, _, _, _, _] := _____`

Quick Review Question 7 Complete the definition of one form of the fourth *walk* bullet: If an ant moves to a cell, the cell continues to have its same amount of chemical. In this case, the current site is initially empty and the cell to the north contains an ant that is facing south. Similar rules apply to ants in the east, south, and west directions facing the current site.

`walk[{a_, _____}, {_, _____}, _, _, _, _, _, _, _, _, _, _, _] := _____`

Walking with Concern for Collision

Quick Review Question 8

- a. Complete the definition of the *walk* rule for the situation in Figure 11.3.1 in which a collision should be avoided for the current ant that faces an empty cell to the north and a northeast ant that faces west. Use the blank symbol (`_`) where possible.

`walk[{a_, _____}, _____, _, _, _, _____, _, _, _, _, _, _] := _____`

- b. Give the number of similar such rules.

Quick Review Question 9

- a. Complete the definition of the *walk* rule for the situation in **Error! Reference source not found.** in which a collision should be avoided where the current site is empty but ants to the north and east both want to move into the site. In this situation, if the amount of chemical at the site is positive, it decrements by 1. Use the blank symbol (`_`) where possible.

```
walk[{a_, _____}, _____, _____, __, __, __, __, __, __, __, __, __] :=
  {Max[a - 1, 0], _____}
```

- b. Give the number of similar such rules.

Visualizing the Simulation

Quick Review Question 10 Suppose *graphList* is a list of square grids representing frames of the simulation at consecutive time steps; and *maxChem* is the maximum amount of chemical in any cell of any grid in *graphList*.

- Write a statement to assign to *g* the *k*th grid, or square matrix, in *graphList*.
- Recall that each cell of a grid is an ordered pair consisting of the amount of chemical and *EMPTY* = "T" or a direction (*NORTH* = "N", *EAST* = "E", *SOUTH* = "S", or *WEST* = "W"). Assign to *siteChem* the first component, the amount of chemical, in the *i-j* cell of *g*.
- Assign to *siteColorAmt* the amount of chemical component, *siteChem*, scaled to be between 0.5 and 1.0, as described in this section.
- Give an expression for the second component, *EMPTY* or a direction, in the *i-j* cell of *g*.
- Write an *If* statement to return an *RGBColor* designation for a cell. If the cell is empty, return a shade of gray (equal amounts of red, green, and blue) based on *siteColorAmt*. Otherwise, return a shade of red based on *siteColorAmt*.

Answers to Quick Review Questions

- `Random[Integer, {1, 4}]`
 - `dir[[r]]`
Thus, the following expression returns a random direction, *N*, *E*, *S*, or *W*:
`dir[[Random[Integer, {1, 4}]]]`
 - `grid =`

```
Table[{0,
  If[Random[] < probAnt,
    dir[[Random[Integer, {1, 4}]]], EMPTY}},
  {n}, {n}]
```
 - `Do[`

```
grid[[IntegerPart[n/2] + 1, j]] =
  {IntegerPart[50 * j / n], EMPTY},
```

```
{j, 1, n}]
```

2. `sense[{a_, EMPTY}, _, _, _, _] := {a, EMPTY}`
3.
 - a. `lst = {n, e, s, w}`
 - b. `mx = Max[lst]`
 - c. `posList = Flatten[Position[lst, mx]]`
 - d. `lng = Length[posList]`
 - e. `rndPos = Random[Integer, {1, lng}]`
 - f. `dirIndex = posList[[rndPos]]`
 - g. `{a, dir[[dirIndex]]}`
 - h. `sense[{a_, _}, {n_, _}, {e_, _}, {s_, _}, {w_, _}] :=
Module[{lst, mx, posList, lng, rndPos},
lst = {n, e, s, w};
mx = Max[lst];
posList = Flatten[Position[lst, mx]];
lng = Length[posList];
rndPos = Random[Integer, {1, lng}];
dir = {NORTH, EAST, SOUTH, WEST};
{a, dir[[posList[[rndPos]]]]}
];`
4. `walk[{a_, EMPTY}, _, _, _, _, _, _, _, _, _, _, _] :=
{Max[a - 1, 0], EMPTY}`
5. `walk[{a_, NORTH}, {_, EMPTY}, _, _, _, _, _, _, _, _, _, _] :=
{a + 1, EMPTY}`
6. `walk[{a_, b_}, _, _, _, _, _, _, _, _, _, _, _] := {a, b}`
7. `walk[{a_, EMPTY}, {_, SOUTH}, _, _, _, _, _, _, _, _, _, _] :=
{a, SOUTH}`
8.
 - a. `walk[{a_, NORTH}, {_, EMPTY}, _, _, _, {_, WEST},
_, _, _, _, _, _] := {a, NORTH}`
 - b. 11
9.
 - a. `walk[{a_, EMPTY}, {_, SOUTH}, {_, WEST},
_, _, _, _, _, _, _, _, _] := {Max[a - 1, 0], EMPTY}`
 - b. 5
10.
 - a. `g = graphList[[k]]`
 - b. `siteChem = g[[i, j, 1]]`
or `siteChem = g[[i]][[j]][[1]]`
 - c. `siteColorAmt = 1.0 - (siteChem * 0.5) / maxChem`
 - d. `g[[i, j, 2]]`
 - e. `If[g[[i, j, 2]] == EMPTY,
RGBColor[siteColorAmt, siteColorAmt, siteColorAmt],
RGBColor[siteColorAmt, 0, 0]]`