情報工学実験2

Teaを用いた複雑系シミュレーション

担当教員名：赤嶺有平

実験日 2008/12/01
提出日 2008/12/15

075730G：澤岻千明

課題 1.1

以下の仕様を満たす CA シミュレータを作成せよ。

- 状態：整数値 (0 or 1)
- 状態遷移規則：

  ムーア近傍の状態の合計が奇数の時、着目セルの状態は１に遷移する
  ムーア近傍の状態の合計が偶数の時、着目セルの状態は０に遷移する

task1.tea のソース

```
//セルの状態変数の宣言
cell {
  //状態変数リスト
  int state
};
$neighbor = { [-1,-1], [0,-1], [1,-1],
              [-1, 0],         [1, 0],
              [-1, 1], [0, 1], [1, 1] };

sync(a_cell of $cell) {
  transform(local(a_cell)) {
    local sum=0;
    for_each(nei of $neighbor) {
        sum += @nei.state;
}
  if(sum%2 == 1) {
    a_cell.state = 1;
}else {
    a_cell.state = 0;
}
  }
}
```

課題 2.1

課題として作成した CA シミュレータの初期状態、状態遷移規則などを変更し結果を考察せよ。

初期状態は task1.fld で変更することができる。１行目の数値は領域は何行・何列かを指定することができる。３行目からの数値は、そのセルの状態を示すもので、これを変更することで、初期状態を変えることができる。

task1.tea の１６行目からの記述を書き換えることで状態遷移規則を変更できる。

課題 1.1 のままの状態遷移規則では、行列を奇数にし、中央に１を配置することで途切れることのない綺麗な遷移を見ることができる。

課題 2.2

衝突すると右に曲がるモデルを、50 %の確率で左に曲がるように変更せよ。

rotate_ag.tea のソース

```
cell {
  int cell_state
};

agent simple {
  int dir
};

EMPTY = 0;
CREATE= 1;
DELETE= 2;

$dir_list = { [0,-1], [1,0], [0,1], [-1,0] }; //方向（状態）をベクトルに変換するためのリスト

for_each(a_cell of $cell) {
  if(a_cell.cell_state == CREATE && 5%) {
    new simple(rand() % 4) -> posof(a_cell); //方向 dir の初期値をランダムな 0〜3 の値として simple を生成
  }else if(a_cell.cell_state == DELETE) {
    delete a_cell.simple;
  }
}

for_each(ag of $simple) {
  transform(local(posof(ag), [0,1], $dir_list[ag.dir])) {
    if(!exist(@[0,1].simple)) //正面にエージェントが存在する？
        ag -> [0,1]; //前に進む
    else if(50%) {
        ag.dir = (ag.dir + 1) % 4; //右方向に９０度回転する
    }else {
        ag.dir = (ag.dir + 3) % 4; //右方向に２７０度回転 ＝ 左方向に９０度回転
    }
  }
}
```

課題 2.3

sugar scape モデルのパラメータ（食欲、初期財産、砂糖の再生度など）の変更が結果に与える影響を考察せよ。

食欲

この値を大きいと、行動するのにより多くの財産を消費していくので、消滅しやすくなる。

初期財産

この値が小さいと、砂糖の山から離れたエージェントは早めに消滅していく。また、財産が貯まるのに多少の時間を要するので子供が誕生するのは遅くなる。

砂糖の再生度

この値が小さいと、早めのサイクルで砂糖が再生するので、より多くの財産を貯めることができるようになり子供が増えてゆく。

逆にこの値が大きいと、遅いサイクルで再生することとなる。すると、エージェントは砂糖を得るために円の外側まで行く必要がある。だが、外側に移動したエージェントはは得る砂糖量よりも失う砂糖量の方が多いので、消滅しやすくなる。

課題 2.4

sugar.tea にコメントを付けよ。

sugar.tea のソース

```
//セルの状態変数宣言
cell {
  int max_sugar, //最大砂糖量
  int sugar      //現在の砂糖量
};

//エージェント ant の内部状態宣言
agent ant {
  int view,  //視野範囲
  int eat,   //食欲
  int worth, //財産

  int display
};

//シミュレーション開始時のみ実行される（初期状態設定）
if(time() == 0) {
  for_each (a_cell of $cell) {
    //砂糖の山を生成する
// 5.0 = 砂糖の最大量、 30,20 = 頂点の座標、5.0 = 角度 (小さいほど急)
    d1 = 5.0 - length(posof(a_cell) - [30,20])/5.0;
    d2 = 5.0 - length(posof(a_cell) - [20,30])/5.0;
```

```
    a_cell.max_sugar = (int)max(max(d1, d2), 0.0);
    a_cell.sugar = a_cell.max_sugar; //砂糖の最大量を呼び出す。

    //エージェントを配置する
    if(1%) { // 全領域の１％にエージェントを配置
      if(50%) { // そのうちの半分がこのタイプ
        new ant(4, 1, 10, 0) -> posof(a_cell);//セルにエージェントを配置する
        //ant の引数は，内部状態の初期値を宣言順に表す
        //視野範囲=4，食欲=1，財産=10，display = 0
      }else{    // それ以外はこのタイプ
        new ant(8, 2, 10, 1) -> posof(a_cell);
      }
    }
  }
}

//4 ステップに一回更新される ＝ 砂糖の再生速度
if(time() % 4 == 0) {
  for_each (a_cell of $cell) {
    a_cell.sugar = min(a_cell.sugar+1, a_cell.max_sugar); //砂糖を生産
  }
}

$view_dir = { [1,0], [0,-1], [-1,0], [0,1] }; // 方向 (状態) をベクトルにするリスト

for_each (an_ant of $ant) { // エージェントの動作
  transform(local(an_ant)) {

    an_ant.worth += @[0,0].sugar; // 財産に現在地の砂糖を追加
    @[0,0].sugar = 0;             // 現在地の砂糖を０に

    an_ant.worth -= an_ant.eat;   // 移動して、財産から食欲の分を引く

// 繁殖の処理
    if(an_ant.worth > an_ant.eat*10) { // 財産が食欲の 10 倍以上あるなら
      new ant(an_ant.view, an_ant.eat, 10, an_ant.display) -> [0,0]; // 同じ能力
で、財産を１０持つ子供を生む
      an_ant.worth -= 10; // 子供にあげた分の財産が減る
    }

    max_sugar = -1;
    $move_to = {};

   //視野内で最も砂糖の多い地点を探す
    for_each(dir of $view_dir) {  // ４方向を調べる
      for(i=1; i<an_ant.view; i+=1) { // 視野範囲の分だけ調べる
```

```
        view_site = dir*i;
        sugar = @view_site.sugar;  // 砂糖の場所

// より多い砂糖を見つけた時の動作
        if(max_sugar < sugar) {
          max_sugar = sugar;
          $move_to = { view_site };
        }else if(max_sugar == sugar) {
          $move_to += view_site;
        }
      }
    }

// 砂糖が多い地点のどれかに移動
    an_ant -> $move_to[rand() % sizeof($move_to)];

   //財産がなくなると消滅する
    if(an_ant.worth < 0) {
      delete an_ant;
    }
  }
}

//第２引数の値をコードウインドウに出力する
log(STDOUT, sizeof($ant));
```
