Subject: 情報工学実験II ： レポート提出
From: IKENOYA Katsutoshi <j05002@ie.u-ryukyu.ac.jp>
Date: Wed, 27 Dec 2006 16:30:49 +0900
To: yuhei@ie.u-ryukyu.ac.jp

情報工学実験II
-TEAを用いたmulti agent simulatorの作成-

レポート作成者：池野谷克俊
学籍番号：055702B

課題1
以下の仕様を満たすCAシミュレータを作成してください.

・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

・ソースコード(odd_even.tea)

```
//セルの状態変数の宣言
cell {
//状態変数リスト
int s
};

//ノイマン近傍
$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)) {
//状態遷移規則の記述
local sum=0; //sync内では，ローカル変数にのみ代入できる
for_each(nei of $neighbor) { //ノイマン近傍のセルにアクセス
sum += @nei.s; //合計を求める
}
if(sum % 2 == 1) {//合計に応じて，次の状態を決定する
a_cell.s = 1;
}else {
a_cell.s = 0;
}
}
}
```

・考察
状態遷移規則を以下の様に変更した。
-ムーア近傍の状態の合計を3で割った余りが2の時，着目セルの状態は 1 に遷移する。
-それ以外は着目セルの状態は0に遷移する。

初期状態を以下に示す

```
16 16
state_name s
0000000000000000
0000000000000000
0000000000000000
0000000000000000
0000000000000000
0000000000000000
0000000100000000
0000000000000000
0000000000000000
0000000000000000
0000000000000000
0000000000000000
0000000000000000
0000000000000000
0000000000000000
```

この場合だと開始直後に全てのセルが0になった。
状態が1であるセルが 1 つしかないためである。

次に初期状態を以下の様にしてみた。

```
16 16
state_name s
0000000000000000
0000000000000000
0000000000000000
0000000000000000
0000000000000000
0000000100000000
0000001000000000
0000000100000000
0000000000000000
0000000000000000
0000000000000000
0000000000000000
0000000000000000
0000000000000000
0000000000000000
```

この状態で実行するとあるパターンを周期的に繰り返した。

課題２
衝突すると右に曲がるモデルを改良して，５０％の確率で左に曲がるようにして
ください．

・ソースコード(simple_ag.tea)
```
cell {
int s
};

agent simple {
int dir
};

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

$dir_list = { [0,-1], [1,0], [0,1], [-1,0] };

for_each(a_cell of $cell) {
if(a_cell.s == CREATE && 5%) {
new simple(rand() % 4) -> posof(a_cell);
}else if(a_cell.s == 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))
ag -> [0,1];
else{
if(50%){
ag.dir = (ag.dir + 1) % 4;
}else{
ag.dir = (ag.dir + 3) % 4;
}
}
}
}
```

課題３
sugar scapeモデルのパラメータの変更が結果に与える影響を調べる
様々なパラメータ（食欲，初期財産，砂糖の再生度など）を変更して，
結果に与える影響を考察してください．

(1)初期財産、砂糖の再生度は変更せずに、食欲を初期財産と同じ10にしてみた。
そうすると、一回行動すると財産が今食べた砂糖の量(5以下)になり、二回目の行動
で全ての蟻が全滅した。
食欲を下げていくと、
"初期財産を10、砂糖の再生度を4ステップ毎にするならば、食欲は4未満にしな
いと蟻は全滅する。"
ということが分かった。

(2)蟻が全滅しないために最低限必要な初期財産を調べてみた。ただし、食欲、
砂糖の再生度は
変更していない。
食欲と砂糖の再生度を変更しないと、初期財産を0にしても蟻は全滅しなかった。

(3)蟻が全滅しないために最低限必要な砂糖の再生度を調べてみた。ただし、食
欲、初期財産は
変更していない。
砂糖の再生度は47ステップ毎までは耐えられた。48ステップ毎にすると蟻は絶滅
した。

以上の結果より、食欲は大きい程全滅する可能性が大きくなる。
砂糖の再生度もステップ数を大きくすると全滅する可能性が大きくなる。
初期財産に関しては食欲と砂糖の再生度の数値によって、絶滅するかどうか
の可能性が変化すると考えた。

課題４
sugar scapeモデルのソースコードを理解する
sugar.teaにコメントを付けてください．

・ソースコード(sugar.tea)
```
cell {
int max_sugar, //最大砂糖量
int sugar //砂糖の量
};

agent ant {
int view, //視野
int eat, //食欲
int worth, //財産

int display
};

//開始時の状態を決定
if(time() == 0) {
for_each (a_cell of $cell) {

//砂糖量を計算する。
d1 = 5.0 - length(posof(a_cell) - [30,20])/5.0; //[30,20]に近い程砂糖が多い
d2 = 5.0 - length(posof(a_cell) - [20,30])/5.0; //[20,30]に近い程砂糖が多い

//砂糖量の決定(d1, d2, 0.0の内大きい方を返す)
a_cell.max_sugar = (int)max(max(d1, d2), 0.0);
a_cell.sugar = a_cell.max_sugar;
```

```
//蟻の生成
//1%の確率で生成
if(rand() % 100 == 0) {
if(rand() % 10 < 5) {
//50%の確率で生成
new ant(4, 1, 10, 0) -> posof(a_cell);
}else{
//50%の確率で生成
new ant(8, 2, 10, 1) -> posof(a_cell);
}
}
}
}

//4ステップ経過した時
if(time() % 4 == 0) {
for_each (a_cell of $cell) {

//注目しているセルの砂糖の量に１足したものと、最大砂糖量の小さい方を返す。
//つまり砂糖の量が最大砂糖量を超えないようにしている。
a_cell.sugar = min(a_cell.sugar+1, a_cell.max_sugar);
}
}

//進行方向の配列(右,上,左,下)
$view_dir = { [1,0], [0,-1], [-1,0], [0,1] };

for_each (an_ant of $ant) {
transform(local(an_ant)) {

//現在のセルにある砂糖を全て食べる(財産にする)
an_ant.worth += @[0,0].sugar;

//食べたので砂糖の量を0にする。
@[0,0].sugar = 0;

//食欲の分だけ財産を消費
an_ant.worth -= an_ant.eat;

//財産が食欲の10倍を超えると新しい蟻を生成
if(an_ant.worth > an_ant.eat*10) {
//view,eatを親の蟻から受け継ぎ現在地に新しい蟻を生成
new ant(an_ant.view, an_ant.eat, 10, 0) -> [0,0];
//新たに生成した蟻に財産(10)を分与したため財産から10引く
an_ant.worth -= 10;
}

//下のfor文で砂糖の量の大小を判断するため初期化
max_sugar = -1;

//下のfor文で使用する配列を生成
$move_to = {};

for_each(dir of $view_dir) {

//前後左右*veiwの範囲のセルの砂糖の量を調べ
//範囲内において砂糖の量が最大の場所の配列を作成
for(i=1; i!=an_ant.view; i+=1) {
view_site = dir*i;
sugar = @view_site.sugar;

if(max_sugar < sugar) {
//現在調べているセルの砂糖量が最大砂糖量より大きければ、
//配列move_toに現在調べている場所を上書きする。
//前の時点での配列の要素は消える。
max_sugar = sugar;
$move_to = { view_site };
}else if(max_sugar == sugar) {
//現在調べているセルの砂糖量と最大砂糖量が等しければ、
//配列move_toへ現在調べている場所を追加で格納
$move_to += view_site;
}
}
}
//move_to配列の要素をランダムで選択し、移動。
an_ant -> $move_to[rand() % sizeof($move_to)];
//財産が無くなると蟻が消滅する。
if(an_ant.worth < 0) {
delete an_ant;
}
}
}

log(STDOUT, sizeof($ant));
```