世界モデル実習（SR）・人間の安全保障実験実習Ⅲ（GR）　　2006 年度夏学期

第十一回 研究道具として使うために

●前回の復習と今回の趣旨

○ 前回は、ユーザー定義関数を使って自分独自の関数を作ることを勉強しました。自由に関数を作ることによって、ルールをシンプルに管理することができます。

○ いよいよ artisoc の勉強も最後になります。（いい加減、飽きてきた？）今日は由緒あるモデルを作って、その楽しい動きを見てみようと思います。

○ 勉強もします。artisoc を研究の道具として使うために、幾つか重要な技法があります。ひとつは、モデルを作成したときに、それが思ったとおりに動いているかどうかをチェックする方法です［**デバッグ**］。もうひとつは、モデルを試行したときの結果をファイルの形で出力する方法です［**ファイル出力**］。最後に、初期状態をファイルの形で artisoc の外から与えてやる方法です［**ファイル読み込み**］。

☆ファイル出力
☆デバッグ
☆ファイル入力

○まず、第８回で作成した「文化伝播モデル（簡易版）」を使って、「ファイル出力」を勉強します。つぎに、今回のモデルを作成して、「デバッグ」と「ファイル読み込み」を勉強することにします。モデルは、山影研究室の HP からダウンロードしてください。

●［第一工程］モデルを準備する

○ ファイルの形で結果を出力すれば、そのファイルを使って（表計算ソフトなどで）自由にグラフや表に編集したり、統計処理をほどこしたりして、検討することができます。

○ 確率を利用することが多いシミュレーションという技法では、同条件で実行したからといって同じ結果になるとは限りません。結果を分析するために何度も実行して傾向をみるという方法を用います。

○ 今回は、＜世界の広さ＞と＜形成される文化の多様性＞の関係を検討してみます。

○ ダウンロードした「文化伝播モデル（簡易版）」は、世界の広さを自由に変えることができ、文化の数をカウントすることができるようになっています。（数え方については第８回レジュメを参照してください）

○ 試しにファイル出力を設定してみます。「設定＞出力設定」で出力種類として「ファイル出力」を選択して「追加」をします。「ファイル出力設定」の画面があくので、「出力名（＝ResultsOutput）」「ファイル名（＝Num_Culture）」を（適当に）入力して、「ファ

イル出力要素」を以下のように追加します。

○　　＜要素名＞　　＜出力値＝＞

「空間幅」　　「GetWidthSpace(Universe.Realm)」

「空間高」　　「GetHeightSpace(Universe.Realm)」

「文化の数」　　「Universe.Num_Culture」

これで、各ステップごとの「空間の幅」「空間の高さ」「文化の数」がファイルの形で出力されることになります。

○ 実行してみましょう。そして適当なところで終了しましょう。（実行するとモデルの横に）さきほどのファイル名のついたファイルが出来ているはずです。そのファイルを表計算ソフトで開けてみて下さい。各ステップでの「空間幅」「空間高」「文化の数」が記録されているのが分かると思います。ではこの結果は捨てておきましょう。

○ ではファイル出力を使って実験結果をまとめてみましょう。「出力設定」でファイル出力（ResultOutput）を編集します。出力間隔を（毎ステップの結果は不要なので）1000 ステップとします。次に「実行環境設定」を開けます。シミュレーションの最大ステップ数 3000 としましょう。3000 ステップでモデルは自動的に終了します。同じ条件で３回実験を繰り返すものとします（本当はもっとたくさんが２０回以上が望ましいですが、時間の都合です）。「連続実行」のタブを選んで最大実行数を３回としましょう。

○ 世界の広さをいろいろ変えて、ファイル出力してやれば、いろいろな条件で繰り返した結果をファイルのかたちで入手することができます。ファイル名を変えずに実行すると、行った実験の結果が同じファイルにどんどん累積で書き加えられていきます。（モデルを実行するまえに、表計算ソフトで開けたファイルは閉じておいてくださいね。）

●　［第二工程］モデルの準備をする

○ では、ライフゲーム（Conway's Game of Life）を作りましょう。このモデルは、1970 年にイギリスの数学者のコンウェイによって考案されたものです。やはり最初のころは手計算で処理されていたそうです。（大変だったでしょうね）本当にシンプルなルールからびっくりするような全体のふるまいが生み出される衝撃は、今でも新鮮です。

○ 空間（大きさは 20×20 とします。名称は Field）とエージェント（名称は Life）を準備しましょう。生命エージェントにコンディションを表す変数（名称 Condition、整数型）を追加します。生命は生きている（Color_Red）か、死んでいる（Color_White）かの二つの値をとるものとします。

○出力設定（マップ出力）をしておきましょう。エージェントの色は変数指定（Condition）にしておいてください。

●［第三工程］生命を作って並べておく

○空間の広さだけ、エージェントを作成して、作り出しておきましょう。(Univ_Init のル

ールです)

```
dim i as integer
dim Lives as Agtset
For i = 0 to GetWidthSpace(Universe.Field)*GetHeightSpace(Universe.Field)-1
        CreateAgt(Universe.Field.Life)
Next i
MakeAgtset(Lives, Universe.Field.Life)
RandomPutAgtsetCell(Lives, False)
```

○生命のコンディションの初期状態を適当に与えておきましょう。まずは、だいたい 20%の生命は生きていて、80%の生命は死んでいることにしましょう。出来たら、試しに出力してみましょう。保存も忘れずにね。（Agt_Init のルールです）

```
If rnd() < 0.2        then
  My.Condition = Color_Red
Else
  My.Condition = Color_White
End if
```

●［第四工程］生命に行動ルールを与える

○生命に行動ルールを与えます。ライフゲームのルールはいたってシンプルです。周囲８マスに生きている生命が幾つあるかを数えます。生きている生命は、ほかに生きている生命が２つか３つあれば生きて続けることができます。周囲に生きている生命が３つあれば死んでいる生命も生き返ります。他の条件では、過密か過疎のため生命は死にます。

☆次ステップにおける生（○）死（×）の条件

		周囲の生きているセルの数							
		1	2	3	4	5	6	7	8
現在の状態	生きている	×	○	○	×	×	×	×	×
	死んでいる	×	×	○	×	×	×	×	×

○上記のようなふるまいをルールとして書いてみましょう。（自力で。。）

☆注意する点・ヒントを以下に書いておきます。

・周囲を認識させるのには、MakeAllAgtsetAroundOwnCell（変数名、視野、含不含）が便利です。

・参照するのは、周囲の生命の現在の Condition ではなく、前ステップの状態にする必要

があります。GetHistory を使いましょう。
・それゆえに、Condition には記憶数を設定しておく必要があります。
・生きている生命の数を数えるときに初期化が必要になります。エラーが出てから修正すれば大丈夫です。

```
Dim Neighbors as Agtset
Dim Neighbor as Agt
Dim Alive as Integer

MakeAllAgtsetAroundOwnCell(Neighbors, 1, False)
Alive = 0
For Each Neighbor in Neighbors
        If GetHistory(Neighbor.Condition, 1) == Color_Red then
                Alive = Alive+1
        End if
Next Neighbor

If Alive == 2 and My.Condition == Color_Red          then
        My.Condition = Color_Red
Elseif  Alive == 3          then
        My.Condition = Color_Red
Else
        My.Condition = Color_White
End if
```

●［第五工程］厳密に検討する

○ さて、無事にライフゲームは動いたでしょうか。作成したモデルが動いているように見えても、本当に意図したとおりに動いているかどうかは分かりにくいです。とくに研究目的でモデルを使用するときには、これを厳密に検討しておく必要があります。

○ モデルを正確に作る一つの重要なコツは、一度に全部作ってしまうのではなく、少しずつ作成しては、その部分が意図通りに動いているかどうかを確認していくという作業を繰り返すことです。経験的には、このほうが早くモデルが完成することが多いようです。

○ モデルが、正確に動いているかどうかをチェックするのに、今回はコンソール画面を使う方法を学びます。print()、println()といった関数でコンソール画面にメッセージを出す方法は学んだと思います。()内に、変数や "文字列" を&でつないでおけば、その値をコンソール画面に出力してくれます。println は、加えて改行もする関数です。

○ この機能を用いて、生命に自分の周りにいる生命の状態と自分の状態や意思決定の様子を我々に報告してもらうことにします。その内容を検討すれば、彼らが我々の意図どおりに意思決定しているかどうかが分かるわけです。

○ 今回は、私の回答例をここに書いておきます。ルールを読んでみて、何をしているのかが分かったら、自分のライフゲームのモデルに書き込んでみてください。全ての生命が報告したら、たいへんなことになってしまうので、ある一つの生命にだけ報告してもらっています。

```
If My.ID == 0      then
  println(GetCountStep()&": My pre-Condition is "&GetHistory(My.Condition, 1)&".")
  println(GetCountStep()&": "&Alive&" Neighbors were alive.")
  println(GetCountStep()&": My Condition is "&My.Condition&".")
End if
```

○ルールが書けたらステップ実行しながらこの生命がちゃんと意図通りに意思決定しているかどうかを確かめてみてください。コンソール画面には、色をあらわす「生の」数字が書き出されます。16777215 は白で、16711680 は赤を表します。そう思ってマトリクス気分で読み取ってください。またコンソール画面は選択してデリートすることができるので、ときどき消してすっきりしてください。

●［第六工程］ファイルを読み込ませる

○ 今まではモデルの初期状態を無作為に確率で与えていました。つぎにファイルの形にしてある初期状態を読み込ませる方法を勉強します。HP に用意している初期値ファイルをダウンロードしてみてください。このなかには有名なライフゲームの初期状態が含まれています。試しにあけてみてください（エクセルでもテキストエディタでも開きます）。各座標と状態の初期値が設定されています。カンマ区切りの CSV 形式と呼ばれるフォーマットです。このファイルはライフゲームのモデルの横に置いてください。

○ Agt_Init のルールで、各エージェントに順番に自分の ID にしたがって X 座標、Y 座標、生死の初期状態を読み込んでもらうことにします。つまり、ID0 の生命には、一番上の X 座標 0、Y 座標 0、対応する状態の値を読み込んでもらうといったかんじですね。まずは、今まで書いていた、＜無作為に状態を与えるルール（AgtInit）＞と＜エージェントをばらまくルール(UnivInit)＞を無効化してください。//を文頭に入れてコメント扱いにしておくと便利です。

○ まずは、ファイルを指定し、開けなくてはなりません。OpenFileCSV(ファイル名, 識別番号, モード)という関数を用います。ファイル名は、ひらくファイルの名称です。”” で囲ってください。今回は GLGlider です。識別番号は、ルールの中で複数のファイルを

扱うときのためにこのファイルに付ける番号です。今回は１つしか開けないので 1 で良いです。モードはそのファイルの扱い方（オープンモード）を決める部分です。1 なら「読み込み」専用のファイルとして扱い、2 なら「書き込み」ファイルとして扱いデータを上書き可能にします、3 なら「追加書き込み」ファイルとして扱い元データの最後に新しいデータを追加可能にします。今回の場合は、読み込み専用ですね。なので、OpenFileCSV(GLGlider, 1, 1)となります。

○ 次にファイル内のデータの読み込みですが、ReadFileCSV(識別番号)です。この関数は、ファイルを開いてから、１回目の実行なら１個目のデータ、７回目なら７個目のデータを読み込みます。具体的に自分の X 座標読み込むルールは以下のようになります。最後に CloseFileCSV(識別番号)でファイルを閉じておきます。

```
dim i as integer
OpenFileCSV(“GLGlider.csv”, 1,1) //まずファイルを開きます
For i = 0 to My.ID*3                    //何番目のデータを X 座標とすべきですか？
        My.X=ReadFileCSV(1)     //データの代入を繰り返します
Next i
CloseFileCSV(1)                         //ファイルを閉じておきます
```

○ 同じように、ファイルを開けて Y 座標と初期状態（生死）も読み込みます。

```
OpenFileCSV("GLglider.csv", 1,1)
For i = 0 to My.ID*3+1
        My.Y=ReadFileCSV(1)
Next i
CloseFileCSV(1)
OpenFileCSV("GLglider.csv", 1,1)
For i = 0 to My.ID*3+2
        My.Condition=ReadFileCSV(1)
Next i
CloseFileCSV(1)
```