# 05. artisocレシピブック
## 通れなくなった道路を迂回しよう

本ドキュメントについてのご質問、『複雑系勉強会』のお問合せは、下記までご連絡ください。

（株）構造計画研究所
社会デザイン・マーケティング部
artisocマーケティング担当　玉田
Tel: 052-222-8461
E-mail: tamada@kke.co.jp

# 火災・浸水・ガレキで道路が通れなくなったら迂回しよう

- 地震が発生すると、道路が通れなくなることが多々あります。
- 道路が通れないとき、避難経路を再探索するルールを追加しましょう。

・歩行モデルの拡張

① 「03. artisocレシピブック」のおさらい
② 道路に線を引く
③ スタートとゴールを設定する
④ 歩行者の行動ルールを変更する
⑤ 画面出力を拡張する
⑥ シミュレーション実行中に「Block」を生成する
⑦ 経路を再探索する
⑧ 立ち止まる

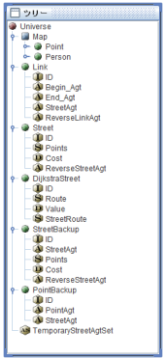
# ①「03. artisocレシピブック」のおさらい

## ■「03. artisocレシピブック」で作成した道路に沿って歩くモデル

- 最短経路を通って目的地まで歩くモデルの動作原理について学びました。

**【Step1】モデルの定義**

- PointエージェントとLinkエージェントを定義して、描画ツールの前準備をします。
- Personエージェントを定義して、最短経路を探索して移動するアルゴリズムを定義します。

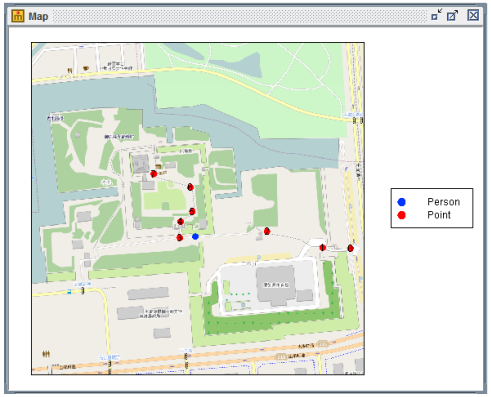


**【Step2】描画ツールで道路を定義**

- 背景画像の道路に沿ってマウスクリックして、ネットワークを定義します。

**【Step3】動きを確認**

- エージェントが移動する様子を確認します。

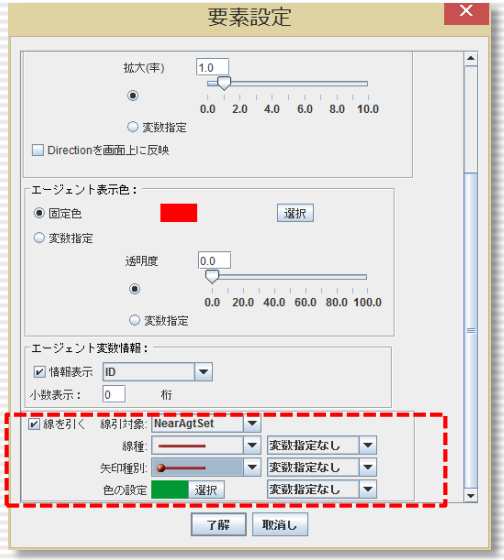


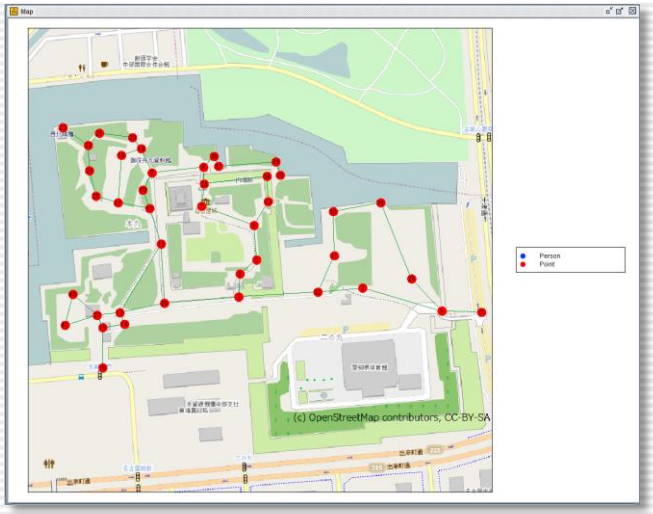

## ■「03. artisocレシピブック」で作成した歩くモデルの問題点と解決策

- 全ての道路が通れることを前提に最短経路を探索します。
- 道路が通れなくなったことを発見し、避難経路を再探索するモデルを作成します。

# ② 道路に線を引く

- つながっている道路に線を引きます。
  - 設定メニューの「出力設定」をクリックして「Map」を選択し、「編集」をクリックします。
  - 「マップ要素リスト」の「Point」を選択し、「編集」をクリックします。
    - 線を引く
      - 線引対象： NearAgtSet
      - 線種： 横棒
      - 矢印種別： 矢印なし
      - 色の指定： 緑色
  - 実行メニューの「ステップ実行」をクリックします。
    - 道路がつながっていることを確認します。



# ③ スタートとゴールを設定する

■スタート地点（Point ID=0）とゴール地点（Point ID=23）を設定します。

● ツリーの「Point」を右クリックして、「ルールエディタ」を選択します。

```
Agt_Step {
    Dim personAgt As Agt

    If My.ID == 0 And GetCountStep()  Mod 10 == 0 Then
        personAgt = CreateAgt(Universe.Map.Person)
        personAgt.X = My.X
        personAgt.Y = My.Y
        personAgt.RouteArray = @dijkstra(My.ID,"23")
        personAgt.RouteCount = 1
End If
}
```

・・・10ステップごとに歩行者を生成する

・・・ゴール地点(Point ID=23)を設定する
（第2引数は文字列なので注意してください）

# ④ 歩行者の行動ルールを変更する

- 経路に沿って進み、目的地に到着したら終了します。
  - ツリーの「Person」を右クリックして、「ルールエディタ」を選択します。
  - 「Agt_Step」の下記の一行を変更します。

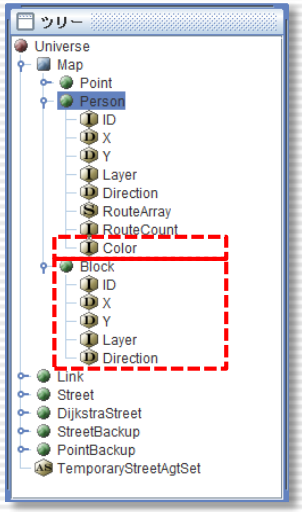
```
If distance > 0 Then
    My.RouteCount = My.RouteCount + 1
    If CountToken(My.RouteArray) > My.RouteCount Then
        targetPointAgt = Universe.Map.Point(CInt(GetToken(My.RouteArray, My.RouteCount)))
        Pursue(targetPointAgt, distance)
    Else
        TerminateAgt(My.UniqueID)
    End If
End If
```

・・・目的地に到着したので、終了する

  - 準備が完了したら、「実行」ボタンをクリックしてください。
    - スタート地点（Point ID=0）からゴール地点（Point ID=23）へ、歩行者が移動する様子が確認できます。
  - 修正したモデルを「05-1.model」として保存します。

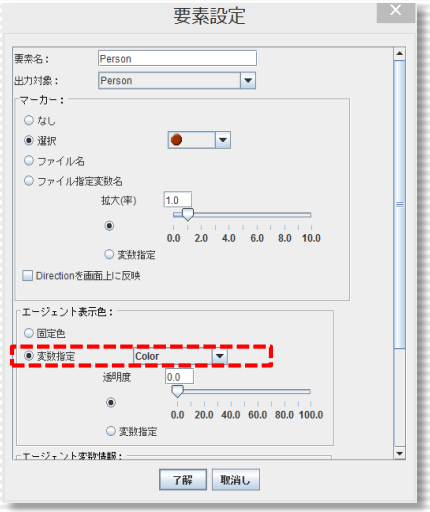
# ⑤ 画面出力を拡張する

■道路の閉塞が確認しやすいように「Block」を追加します。
- 「Universe」の「Map」に「Block」エージェントを追加します。
- 「Block」は、指定した2点間に「×」を表示します。




■歩行者が経路を再探索したことを確認するために表示色を変更します。
- 「Person」に次の変数を追加します。
  - 変数名： Color　　：整数型
    - 表示色を格納します。

■エージェントの表示色を設定します。
- 設定メニューの「出力設定」をクリックして「Map」を選択し、「編集」をクリックします。
- 「マップ要素リスト」の「Person」を選択し、「編集」をクリックします。
  - エージェント表示色
    - 変数指定： Color



# ⑥ シミュレーション実行中に「Block」を生成する（1）

■「Block」を生成します。

- ツリーの「Universe」を右クリックして、「ルールエディタ」を選択します。

```
Univ_Step_Begin {
    create_block(50, 3, 26, 10000)
    create_block(100, 19, 9, 10000)
    create_block(150, 1, 2, 10000)
}
```

・・・任意のステップでコストを更新する

引数は以下の通り

1. 更新するタイミング（ステップ数）
2. Point ID1
3. Point ID2
4. 更新後のコスト

※Point ID1とPoint ID2の間に「Block」を配置します

- 「Street」の「Cost」は、該当の経路の選択しにくさを表しており、通常は距離を与えております。最短経路を選択する際、「Cost」の小さい経路を選択します。
- 上記の例では、「Cost」に「10000」を設定していますが、他の経路の「Cost」と比較して大きければ別の値でも問題ありません。
- 各「Street」の「Cost」を知りたい場合は、「Universe」の「Univ_Step_Begin」で「print_street」関数を実行すると、コンソール画面に表示できます。

```
print_street()
```

# ⑥ シミュレーション実行中に「Block」を生成する（2）

■「Block」を生成します。

● ツリーの「Universe」を右クリックして、「ルールエディタ」を選択します。

```
Sub create_block(countStep As Long, beginPointID As Integer, endPointID As Integer, newCost As Double)
    Dim streetAgt As Agt
    Dim beginPointAgt As Agt
    Dim endPointAgt As Agt
    Dim blockAgt As Agt

    If GetCountStep() == countStep Then
        update_cost(beginPointID, endPointID, newCost)

        beginPointAgt = Universe.Map.Point(beginPointID)
        endPointAgt = Universe.Map.Point(endPointID)
        blockAgt = CreateAgt(Universe.Map.Block)
        blockAgt.X = (beginPointAgt.X + endPointAgt.X) / 2
        blockAgt.Y = (beginPointAgt.Y + endPointAgt.Y) / 2
    End If
}
```

・・・隣り合う2点が含まれるStreetのCostを更新する

・・・道路閉塞（×）を2点の間に表示する

# ⑦ 経路を再探索する(1)

■「Block」が配置された場所に差し掛かったとき、経路を再探索します。

● ツリーの「Person」を右クリックして、「ルールエディタ」を選択します。

```
Agt_Step{
    Dim targetPointAgt As Agt
    Dim distance As Double
    Dim lastTargetPointAgt As Agt
```

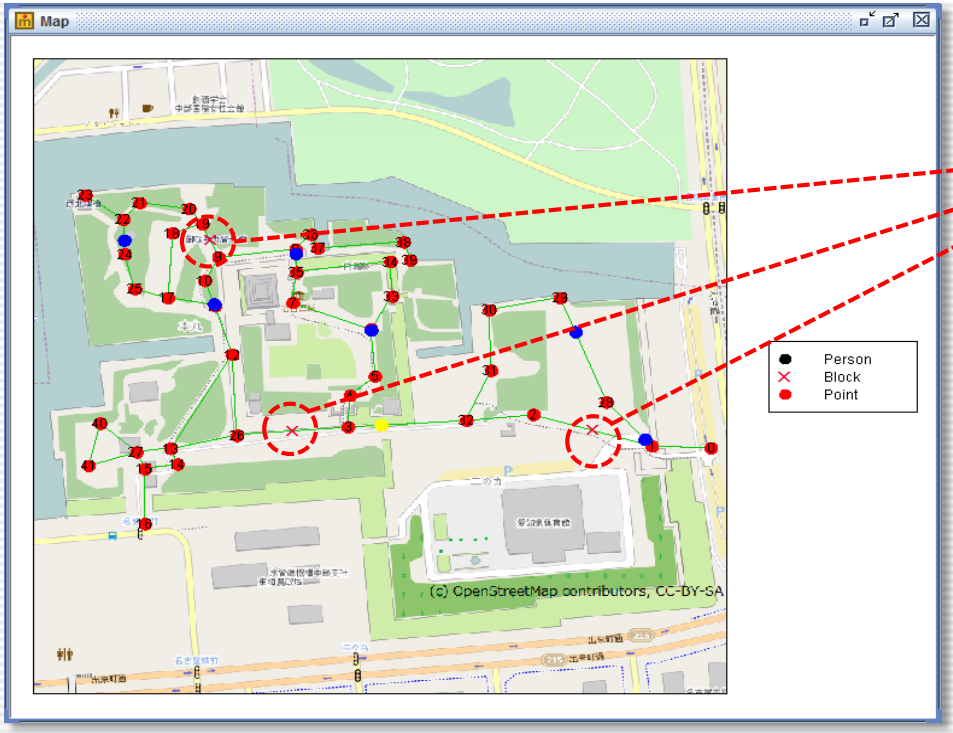
```
If distance > 0 Then                                          ・・・Pointに到着したとき
    lastTargetPointAgt = Universe.Map.Point(CInt(GetToken(My.RouteArray, My.RouteCount)))
    My.RouteCount = My.RouteCount + 1
    If CountToken(My.RouteArray) > My.RouteCount Then
        targetPointAgt = Universe.Map.Point(CInt(GetToken(My.RouteArray, My.RouteCount)))
        streetAgt = @get_street_from_points(lastTargetPointAgt.ID, targetPointAgt.ID)
        If streetAgt.Cost == 10000 Then                       ・・・StreetのCostが10000の場合は
            reroute(targetPointAgt)                               経路を再探索する
        Else
            Pursue(targetPointAgt, distance)                  ・・・余剰分、次のPointへ向かう
        End If
```

# ⑦ 経路を再探索する(2)

```
Sub reroute(targetPointAgt As Agt)
{
    Dim startPointAgt As Agt
    Dim goalPointID As Integer
    Dim newRoute As String
    Dim lastRouteArray As String
    Dim i As Integer

    startPointAgt = Universe.Map.Point(CInt(GetToken(My.RouteArray, My.RouteCount - 1)))
    goalPointID = CInt(GetToken(My.RouteArray, CountToken(My.RouteArray) - 1))
    newRoute = @dijkstra(startPointAgt.ID, CStr(goalPointID))          ・・・現在地点から経路を再探索する

    lastRouteArray = My.RouteArray
    My.RouteArray = ""
    For i=0 To My.RouteCount - 2                                       ・・・これまでに通った経路を格納する
        My.RouteArray = My.RouteArray & GetToken(lastRouteArray, i) & ","
    Next i
    My.RouteArray = My.RouteArray & newRoute                           ・・・新しい経路を追加する

    My.Color = COLOR_YELLOW                                            ・・・エージェントの色を変更する
}
```

# ⑦ 経路を再探索する(3)

- モデルを実行し、Blockを回避した経路を通るか確認します。
  - 実行メニューの「実行」をクリックします。
  - 修正したモデルを「05-2.model」として保存します。




  - このモデルでは、Costが10000に更新されたStreet以外に通れる道がない場合は通ります。
    - Costが10000のStreetは通れないようにするにはどうすればよいか考えてみましょう。

# ⑧ 立ち止まる（1）

■歩行者が移動する前にStreetを取得して判定します。

●ツリーの「Person」を右クリックして、「ルールエディタ」を選択します。

```
Agt_Step {
    Dim targetPointAgt As Agt
    Dim distance As Double
    Dim lastTargetPointAgt As Agt
    Dim streetAgt As Agt

    lastTargetPointAgt = Universe.Map.Point(CInt(GetToken(My.RouteArray, My.RouteCount-1)))
    targetPointAgt = Universe.Map.Point(CInt(GetToken(My.RouteArray, My.RouteCount)))
    streetAgt = @get_street_from_points(lastTargetPointAgt.ID, targetPointAgt.ID)
    If streetAgt.Cost == 10000 Then
        My.Color = COLOR_GREEN
    ElseIf My.RouteCount < CountToken(My.RouteArray) Then
```

・・・立ち止まる

■全ての歩行者がCostが10000のStreetを通るように設定します。

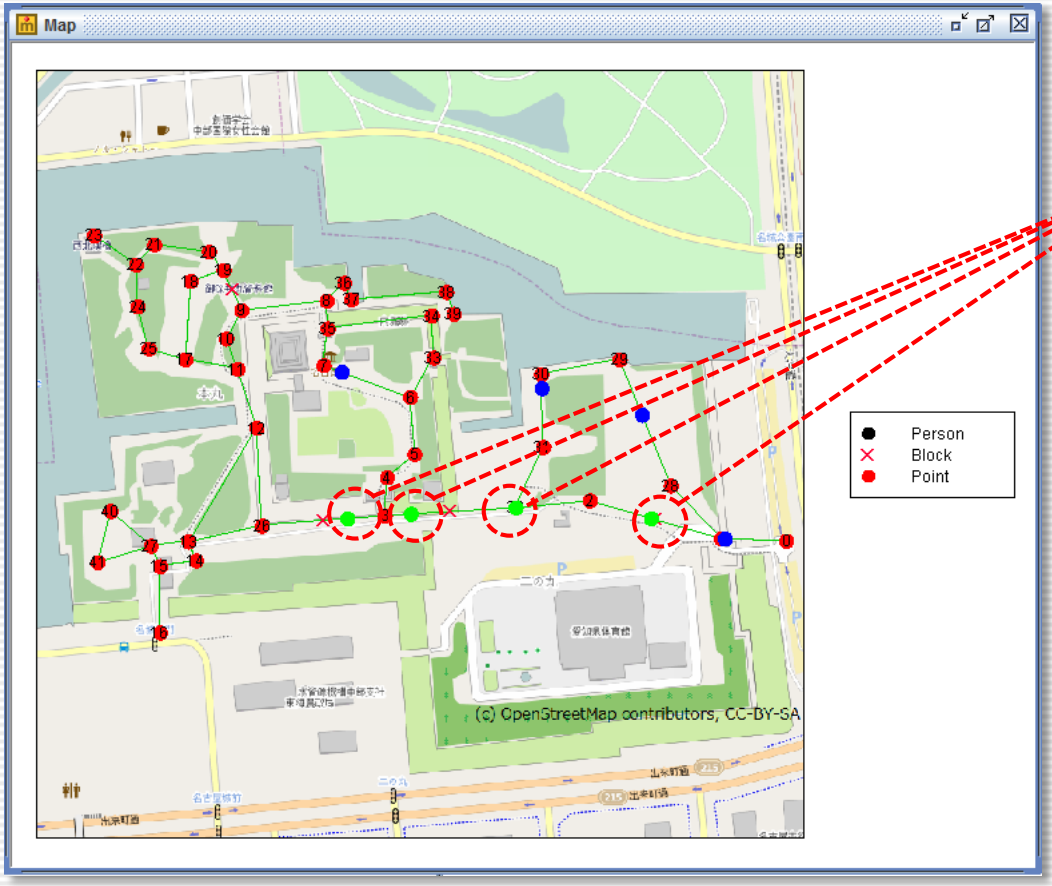
●ツリーの「Universe」を右クリックして、「ルールエディタ」を選択します。

```
create_block(200, 32, 3, 10000)
```

# ⑧ 立ち止まる（2）

■モデルを実行し、歩行者が立ち止まるか確認します。

- 実行メニューの「実行」をクリックします。
- 修正したモデルを「05-3.model」として保存します。




歩行者が立ち止まる