

03. artisocレシピブック

ダイクストラ法を使って、最短経路を自動的に探索しよう

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

最短経路を探索しよう

- マップが複雑になると、歩く経路を指定することは手間がかかります。
- ダイクストラ法を使って、自動的に最短経路を探索します。

・歩行モデルの拡張

- ① 「02. artisocレシピブック」のおさらい
- ② ダイクストラ法とは？
- ③ 歩行モデルの拡張
- ④ Universeで計算ライブラリを初期化
- ⑤ 複数の歩行者を生成
- ⑥ 歩行者の行動ルールを変更
- ⑦ 道路を拡張



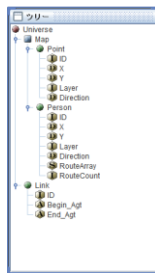
①「02. artisocレシピブック」のおさらい

■「02. artisocレシピブック」で作成した道路に沿って歩くモデル

- 描画ツールで道路を作成していく方法、歩くモデルの動作原理について学びました。

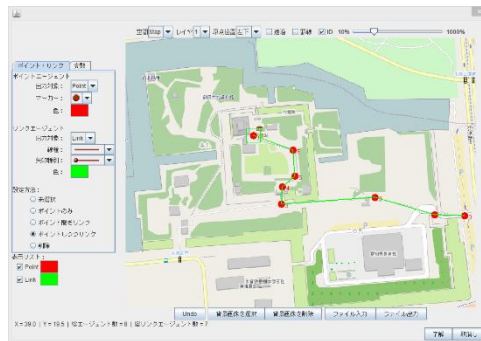
【Step1】モデルの定義

- 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、指定した経路で移動するアルゴリズムを定義します。



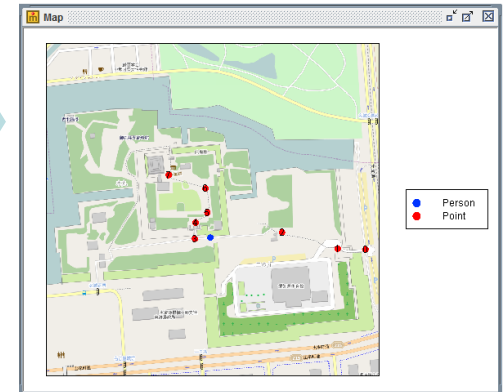
【Step2】描画ツールで道路を定義

- 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

- 指定した順番にPointを通してゴールにたどり着くことを確認します。

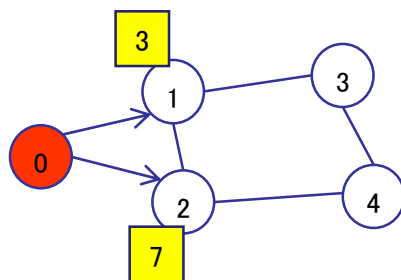
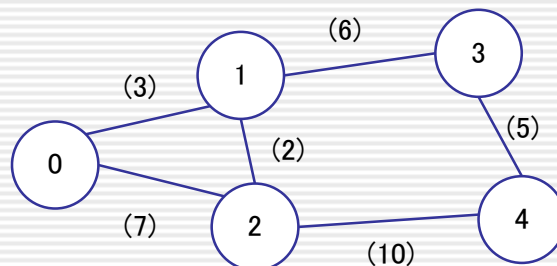


■「02. artisocレシピブック」で作成した歩くモデルの問題点と解決策

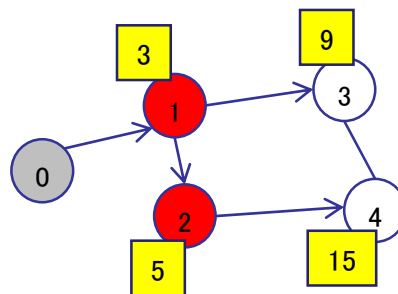
- 道路ネットワークが単純なときは手作業で経路を指定できますが、複雑になると大変面倒です。
- そこでダイクストラ法を使って、最短経路を自動的に探索するモデルを作成します。

② ダイクストラ法とは？

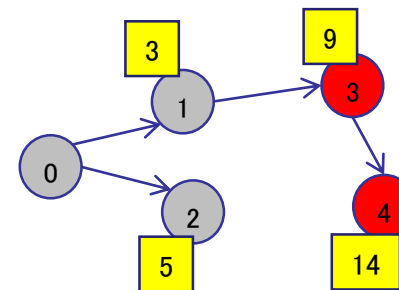
■ 下図のネットワークで、点0からのコストを算出します。(カッコ内は各リンクのコスト)



- Point:0に接続するリンク $0 \rightarrow 1$, $0 \rightarrow 2$ を抽出します。
- Point:1, 2のコストを算出します。



- Point:1に接続するリンク1→2, 1→3を抽出します。
- Point:2に接続するリンク2→1, 2→4を抽出します。
- Point:2のコストは、0→1→2を通った場合に最少となるので、5に更新します。
- Point:3, 4のコストを算出します。



- Point:3に接続するリンク3→4を抽出します。
- Point:4に接続するリンク4→3を抽出します。
- Point:4のコストは、0→1→3→4を通った場合に最少となるので、14に更新します。

■点0から各点への最短経路は、矢印の向きをたどることで求めることができます。

●ダイクストラ法の詳細については、[Wikipedia](#)で調べてみましょう。

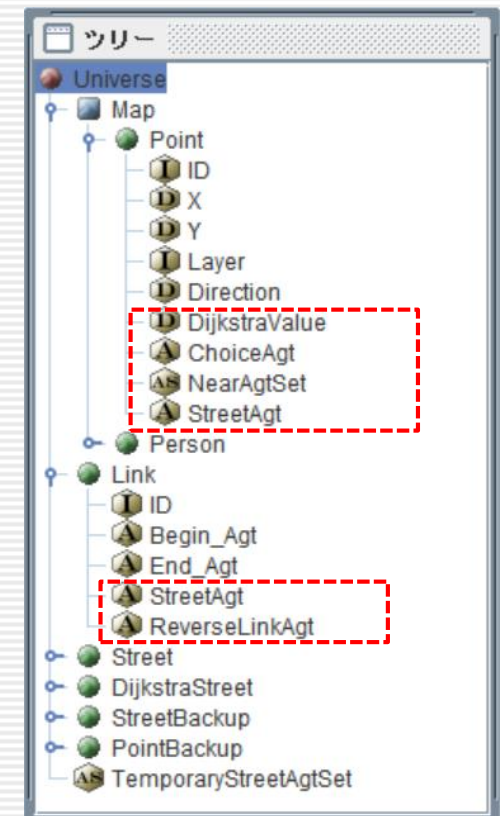
③ 歩行モデルの拡張(1)

■ 交差点や端点の間のPointを束ねて、Streetで管理します。

- 描画ツールで定義したLinkの情報を元に、交差点や端点の間のLinkを束ねて、Streetを生成します。

■ 「Point」と「Link」を拡張します。

- 「Point」に次の変数を追加します。
 - 変数名: DijkstraValue :実数型
 - 計算値を一時的に格納します。
 - 変数名: ChoiceAgt :エージェント型
 - 最短経路のリンクを一時的に格納します。
 - 変数名: NearAgtSet :エージェント集合型
 - 接続しているPointを格納します。
 - 変数名: StreetAgt :エージェント型
 - 属しているStreetを格納します。
- 「Link」に次の変数を追加します。
 - 変数名: StreetAgt :エージェント型
 - 計算値を一時的に格納します。
 - 変数名: ReverseLinkAgt :エージェント型
 - 逆方向のLinkを格納します。



③ 歩行モデルの拡張(2)

■「Street」と「DijkstraStreet」を追加します。

- 「Universe」に「Street」エージェントを追加します。
Link情報から「Street」を生成して保持します。

- 変数名: Points : 文字列型

- 属しているPointのID配列です。

- 変数名: Cost : 実数型

- コストを格納します。

- 変数名: ReverseStreetAgt : エージェント型

- 逆方向のStreetを格納します。

- 「Universe」に「DijkstraStreet」エージェントを追加します。
「DijkstraStreet」は、ダイクストラ法の計算結果を一時的に格納します。

- 変数名: Route : 文字列型

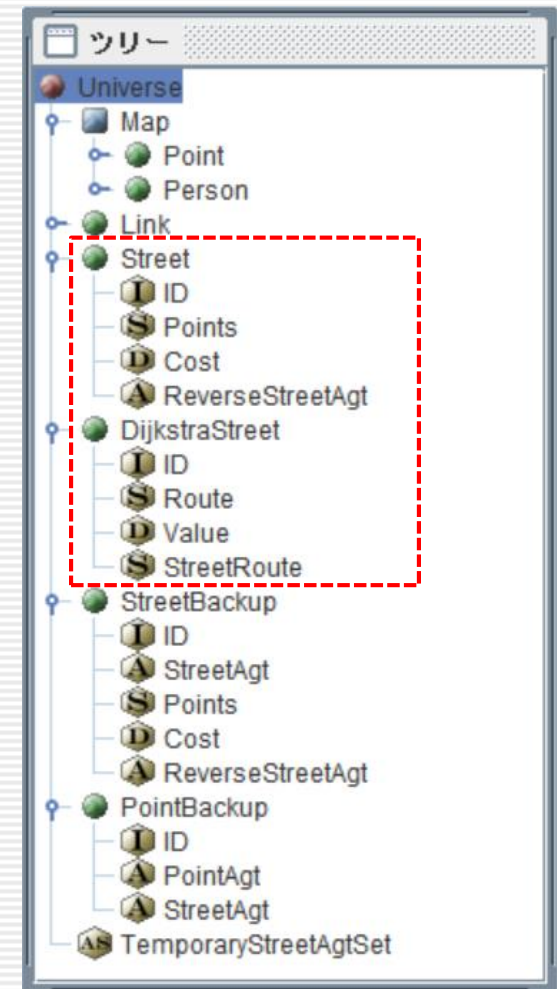
- 当該経路のPointのID配列です。

- 変数名: Value : 実数型

- 当該経路のコストを一時的に格納します。

- 変数名: StreetRoute : 文字列型実数型

- 当該経路のStreetのID配列です。



③ 歩行モデルの拡張(3)

■「StreetBackup」、「PointBackup」、「TemporaryStreetAgtSet」の追加

- 「Universe」に「StreetBackup」エージェントを追加します。
ダイクストラ法で計算する際にStreetが分断されるため、
元の状態をバックアップします。

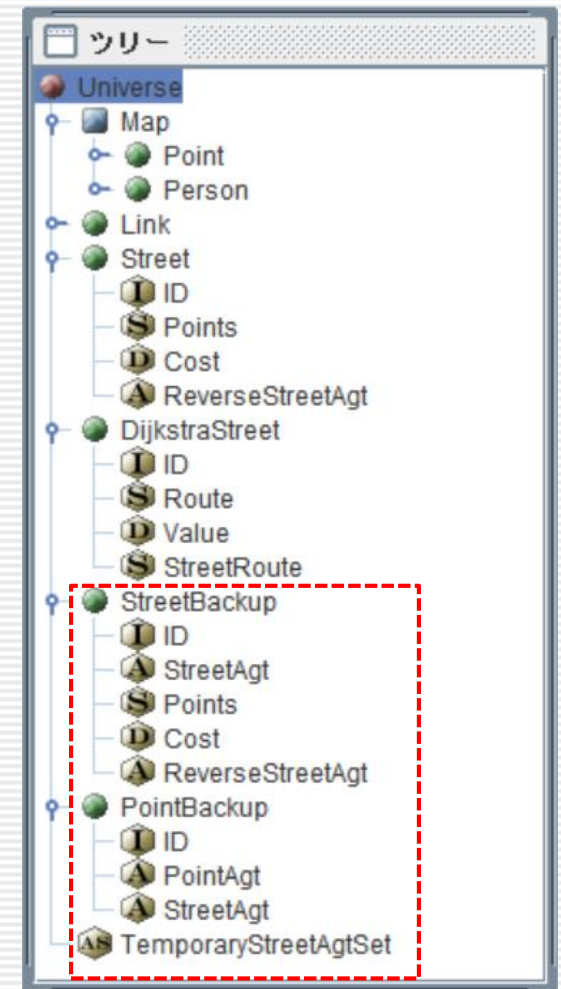
- 変数名: StreetAgt :エージェント型
 - 該当するStreetを一時的に保持します。
- 変数名: Points :文字列型
 - 属しているPointのID配列を一時的に保持します。
- 変数名: Cost :実数型
 - コストを一時的に保持します。
- 変数名: ReverseStreetAgt :エージェント型
 - 逆方向のStreetを一時的に保持します。

- 「Universe」に「PointBackup」エージェントを追加します。

- 変数名: PointAgt :エージェント型
 - 該当するStreetを一時的に保持します。
- 変数名: StreetAgt :エージェント型
 - 属しているStreetを一時的に保持します。

- 「Universe」に「TemporaryStreetAgtSet」を追加します。

- 変数名: TemporaryStreetAgtSet :エージェント集合型
 - 計算前に追加したStreetを一時的に保持します。



④ Universeで計算ライブラリを初期化

- ダイクストラ計算ライブラリ「dijkstra.inc」を利用します。
 - 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_Init」で計算ライブラリを初期化します。

```
include "dijkstra.inc"
```

・・・計算ライブラリを読み込む

```
Univ_Init{
```

```
    initialize_dijkstra()
```

・・・計算ライブラリを初期化する

```
}
```

【ダイクストラ法計算ライブラリの使い方】

- ・ 「dijkstra.inc」は、ダイクストラ法で最短経路を探索するための計算ライブラリです。
- ・ 計算ライブラリを利用するためには、Universeの先頭に「include "dijkstra.inc"」と記述し、「Univ_Init」で、初期化のための関数「initialize_dijkstra()」を実行してください。
- ・ 最短経路を取得するためには、「@dijkstra([始点のPointのID], [候補地のPointのID配列])」の形式で指定します。

⑤ 複数の歩行者を生成

■ 経路に沿って進み、目的地に到着したら新しい経路を設定します。

● 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_Init{

Dim personAgt **As** Agt

Dim nearPointAgt **As** Agt

If My.ID == 0 And GetCountStep() < 5 **Then**

・・・Point ID=0から、4人の歩行者を生成する

 personAgt = CreateAgt(Universe.Map.Person)

 personAgt.X = My.X

 personAgt.Y = My.Y

If CountAgtSet(My.NearAgtSet) > 0 **Then**

 nearPointAgt = GetAgt(My.NearAgtSet, Cint(Rnd() * CountAgtSet(My.NearAgtSet)))

 personAgt.RouteArray = @dijkstra(My.ID, CStr(nearPointAgt.ID))

 personAgt.RouteCount = 1

・・・隣接したPointへの最短経路を取得する
(第2引数は文字列なので注意してください)

Else

 personAgt.RouteArray = CStr(My.ID)

 personAgt.RouteCount = 0

・・・隣接したPointがない場合

End If

End If

}

⑥ 歩行者の行動ルールを変更(1)

■ 経路に沿って進み、目的地に到着したら新しい経路を設定します。

- 新しい経路は2箇所の候補地をランダムで選択し、何れか経路長が短い方を選択します。
- 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_Init {
```

```
}
```

```
Agt_Step {
```

```
    Dim targetPointAgt As Agt
```

```
    Dim distance As Double
```

```
    If My.RouteCount < CountToken(My.RouteArray) Then
```

...経路に沿って進む

```
        targetPointAgt = Universe.Map.Point(CInt(GetToken(My.RouteArray, My.RouteCount)))
```

```
        distance = Pursue(targetPointAgt, 1)
```

...Pursue関数で、
targetPointAgtの方向に進む

⑥ 歩行者の行動ルールを変更(2)

If distance > 0 **Then**

...Pointに到着したとき

My.RouteCount = My.RouteCount + 1

If CountToken(My.RouteArray) > My.RouteCount **Then**

...余剰分、次のPointへ向かう

targetPointAgt = Universe.Map.Point(CInt(GetToken(My.RouteArray, My.RouteCount)))

Pursue(targetPointAgt, distance)

Else

add_new_route(targetPointAgt)

...目的地に到着したので、
新しい経路を設定する

End If

End If

Else

targetPointAgt = Universe.Map.Point(CInt(GetToken(My.RouteArray, My.RouteCount-1)))

add_new_route(targetPointAgt)

...新しい経路を設定する

End If

}

⑥ 歩行者の行動ルールを変更(3)

```
Sub add_new_route(targetPointAgt As Agt)
```

```
{
```

```
    Dim newTargetPointAgt As Agt
```

```
    Dim newTargetPointAgt2 As Agt
```

```
    Dim newRoute As String
```

```
    newRoute = ""
```

...2箇所の候補地をランダムで選択する

```
    newTargetPointAgt = Universe.Map.Point(CInt(Rnd() * CountAgt(Universe.Map.Point)))
```

```
    newTargetPointAgt2 = Universe.Map.Point(CInt(Rnd() * CountAgt(Universe.Map.Point)))
```

```
    If ((targetPointAgt.ID <> newTargetPointAgt.ID) And (targetPointAgt.ID <> newTargetPointAgt2.ID))  
        And (newTargetPointAgt.ID <> newTargetPointAgt2.ID) Then
```

```
        newRoute = @dijkstra(targetPointAgt.ID, CStr(newTargetPointAgt.ID) & "," &  
            CStr(newTargetPointAgt2.ID))
```

...2箇所の候補地のうち、経路長の
短い方の経路を返す

```
    End If
```

```
    If Len(newRoute) > 0 Then
```

```
        My.RouteArray = My.RouteArray & "," & newRoute
```

...2箇所の候補地とも到達できない
場合は""を返す

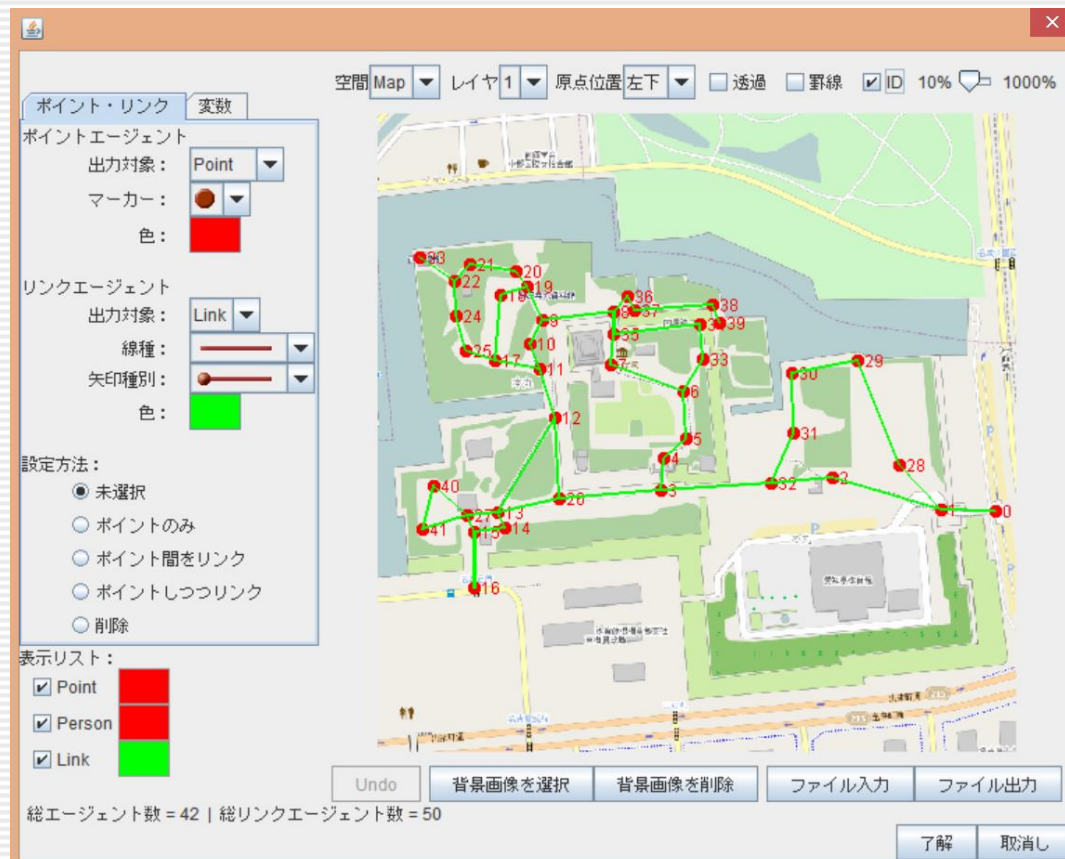
```
    End If
```

```
}
```

⑦ 道路の拡張

■ 道路を拡張します。

- ツリーの「Map」を右クリックして「初期値設定」を選択し、描画ツールを表示します。
- 道路ネットワークを自由に拡張してください。



03.model