ユーザマニュアル

ご注意

（1） 目的の如何にかかわらず、また電子コピー・機械コピー・写真複写・録音などいかなる手段によっても、弊社の書面による許諾なしに本書の内容の一部または全部を複写・転載することを禁じます。

（2） 本マニュアルに記載された内容は予告なく変更されることがあります。

（3） 本書の内容には万全を期しておりますが、万一ご不審な点や誤り、記載漏れなどお気づきの点がございましたら、弊社までご連絡下さい。

（4） 運用した結果の影響につきましては、（3）の項にかかわらず責任を負いかねますのでご了承ください。

（5） 乱丁・落丁はお取り換えいたします。

〒164-0011　東京都中野区中央 4-5-3
株式会社 構造計画研究所
創造工学部　artisoc 係
Tel: (03)5342-1125

2017/03/30

目次

1. artisoc 入門

第 1 章　artisoc 入門

1.1. サンプルを実行する

artisoc には、実際にどのように動くのかを体感してもらうために、サンプルモデルがあらかじめ付属されています。それぞれのモデルファイル（*.model）を開けば、最小限の設定をするだけで、すぐにシミュレーションを開始することができます。
それでは、実際に試してみながら、artisoc の使い方に慣れていくことにしましょう。

準備作業

スタートメニューの「すべてのアプリ」-「artisoc」-「models」-「Mouth-Com」を選択し、mouth-com.model を開いてください。

1.1.1. 口コミモデル

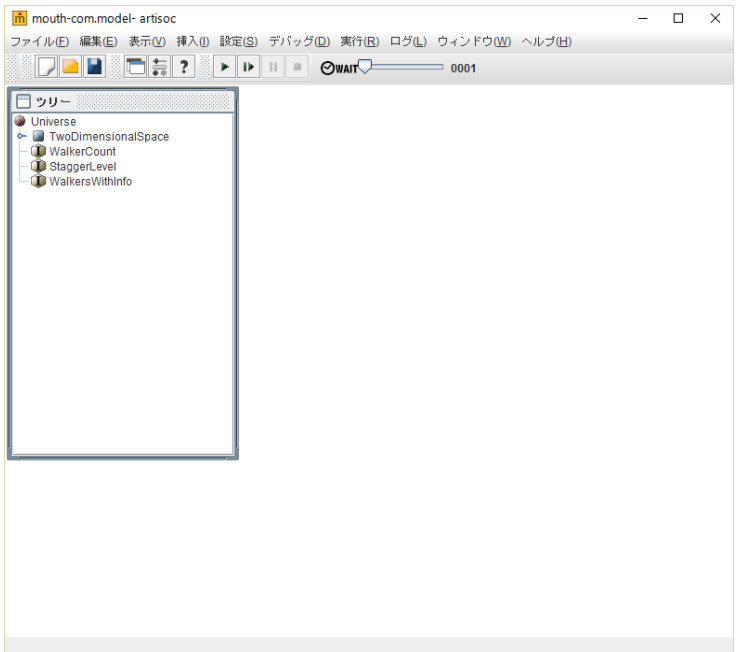
（1） このシミュレーションについて

情報を持った少人数が、口コミで情報を流すことによって、うわさが広まっていく過程をシミュレートするモデルです。
「二次元空間」上を、情報を持った「歩行者」エージェント（赤）がぶらついています。情報を持たないエージェント（緑）とぶつかると情報は伝達され、ぶつかられたエージェントの色は緑から赤へ変わります。次第に赤色のエージェントが増えていき、うわさは広まります。

もちろん、これらの設定はユーザの皆様が自由に変更することも出来ますが、まずは、設定はこのままで実行してみましょう。

（2）各種設定

「mouth-com.model」ファイルを開くと、次の図のようなウィンドウが開きます。



■ツリー：

各コンポーネントを階層構造で表示する、このシミュレータの一番基本的な画面です。ツリーの中には必ず一つの「Universe」が存在します。そしてその下に「空間」や「エージェント」、「変数」などのコンポーネントを定義できます。これらについては後の章で詳しく説明します。

それでは実際に動かしてみましょう。

動作させるには、メニューバー下の実行パネルを使います。

実行パネルは他の空間にドラッグ＆ドロップすることで分離できます。もとに戻したい時は、閉じるアイコンをクリックしてください。

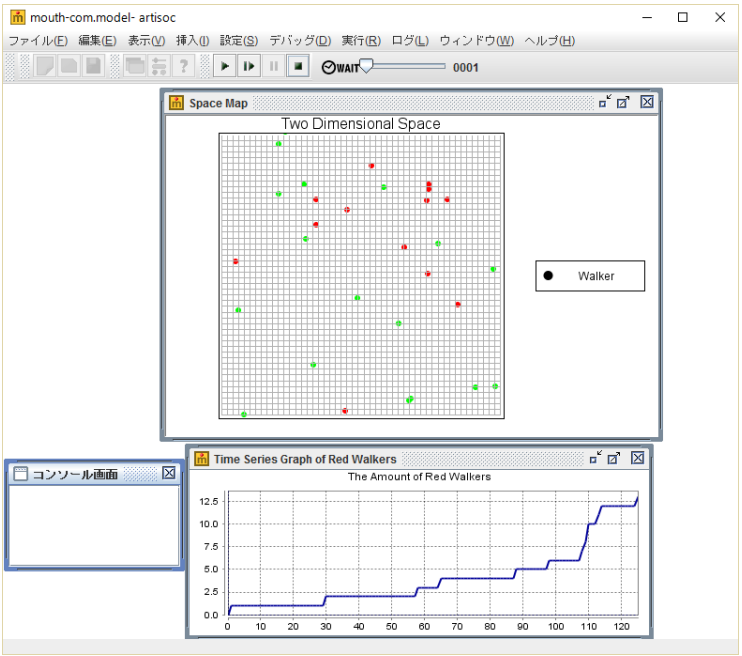

「実行」ボタン　シミュレーションを開始する


「ステップ実行」ボタン　シミュレーションを 1 ステップ分だけ実行


「一時停止」ボタン　シミュレーションを一時停止する


「停止」ボタン　シミュレーションを終了する

さて、それでは実行してみましょう。「実行」ボタンをクリックしてみてください。次のような画面が表示されましたか？



（3）実行結果

このように「ツリー」「Time Series Graph of Red Walkers」「Space Map」「コンソール画面」という 4 つのウィンドウが表示されましたね。これらのウィンドウは「停止」か「一時停止」を押すまで、または全てのエージェントに情報が伝わるまで、シミュレーションを続け、リアルタイムに結果を表示します。

1.1.2. 口コミモデル・・・情報伝播シミュレーション

さて、前項において実際にモデルを実行してみましたが、このままでは一通りのシミュレーションしか実行することが出来ません。ここでは、設定を変更することにより、違う結果を導き出してみましょう。

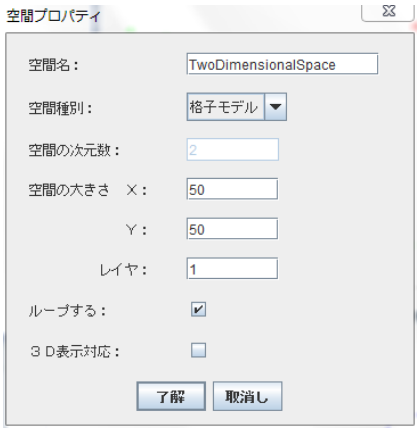
コンポーネントの再設定

空間の大きさを変えてみよう

初期段階では 50×50 の二次元空間上を歩行者たちが移動していましたが、空間の大きさを 100×25 に変えて見ましょう。手順は次の通りです。

「ツリー」上の「空間」選択した状態で、「空間」を右クリックし、メニューから「プロパティ」を選択します。

すると次の画面が表示されます。



「空間の大きさ X」を 100、「空間の大きさ Y」を 25 に変更します。

出力を再設定する

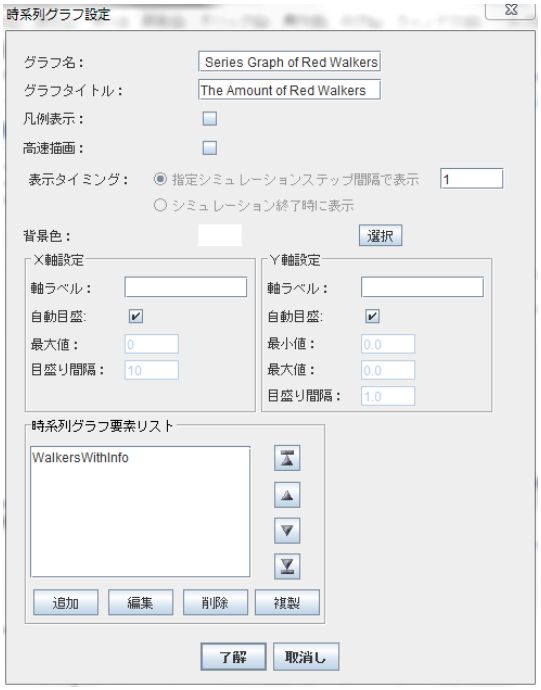
次にここでは、出力に関して色々とアレンジしてみましょう。

「設定」メニューから「出力設定」を選択して、次の画面を表示させます。

（1）時系列グラフの線を太くし、マーカーを追加する

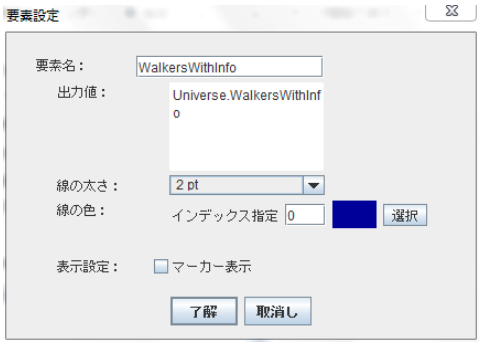
時系列グラフ（人数）の折れ線の頂点部分にマーカーを追加し、線を太くして、グラフを見やすくしてみましょう。手順は次の通りです。

1． リストの中の「Time Series Graph of Red Walkers」を選択し、「編集」ボタンをクリックして、次の時系列グラフ設定ダイアログを表示させます。

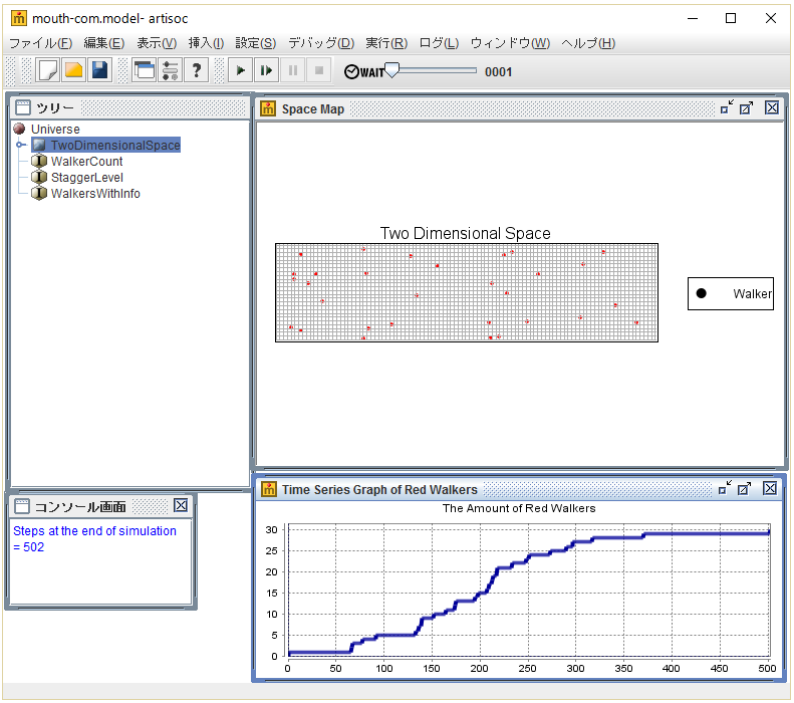


2. 「時系列グラフ要素リスト」の「WalkersWithInfo」を選択し、「編集」ボタンをクリックします。

3. 「線の太さ」を 2pt から 4pt に変更し、「マーカー表示」にチェックを入れ、「了解」ボタンをクリックします。

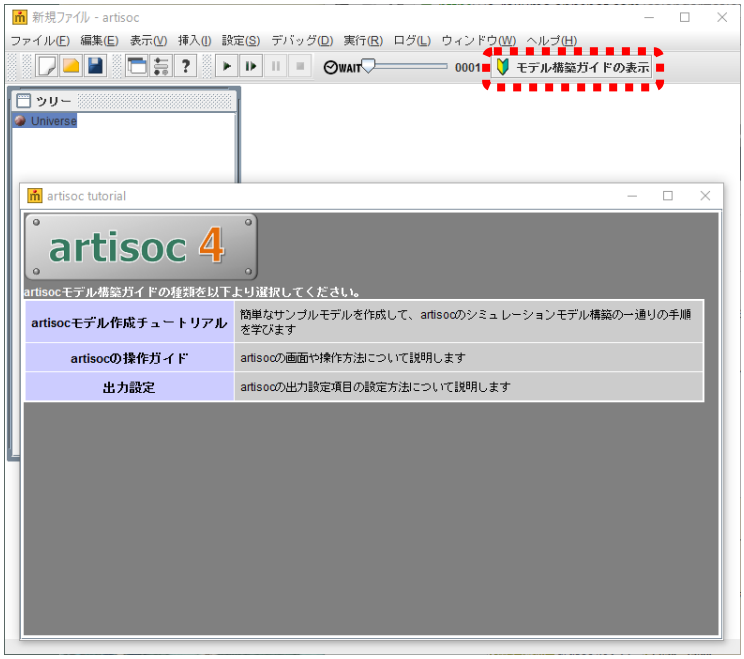


さて再び準備が出来ました。実行パネルから実行ボタンをクリックしてみましょう。



1.2. 簡単なモデルを構築する

次に、まったく白紙の状態からエージェントモデルを構築するための基本的な要素と、手順について学びます。まず、artisocを起動します。新規画面上には、下図のように「モデル構築ガイドの表示」ボタンが表示されています。このボタンを押下すると、モデルを構築するために有用な情報をまとめた数種類のガイドが選択できます。簡単なモデル作成を通して、モデル構築までの一連の手順を「artisocモデル作成チュートリアル」にまとめてありますので、まずはこちらを参照しながら、一通りの手順を学んでみてください。



2. リファレンス

第2章　リファレンス

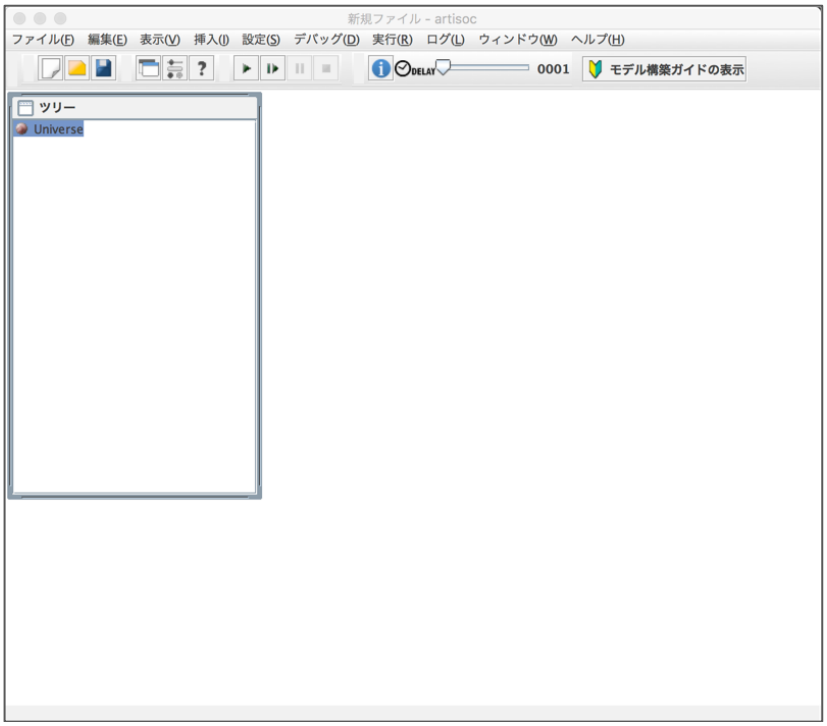
2.1. ライセンス認証

artisoc をご利用いただくためには、発行されたライセンス情報に基づいてライセンス認証を行う必要があります。ライセンス認証を行うためには、認証サーバとの通信を行いますので、お使いのコンピュータがインターネットに接続されている必要があります。最初に artisoc を起動すると、ライセンス認証画面が表示されます。

発行されたライセンスの KeyCode を入力してください。ライセンス情報はお使いのコンピュータ上に保存されますので、都度入力いただく必要はありません。発行されたライセンスで認証できない場合には artisoc 窓口までお問い合わせください。

2.2. 画面説明

artisoc 画面のメニューについて解説します。



メニューバーには、ファイル、編集、表示、挿入、設定、デバッグ、実行、ログ、ウィンドウ、ヘルプがあります。

■ファイル： 2.3 を参照

■編集： ユーザの入力を助ける基本的な操作を行うことができます。

■表示： 各項目について表示、非表示を切り替えます。

■挿入： 2.3 を参照

■設定： 2.4 を参照

■デバッグ： 2.9 を参照

■実行： 2.5 を参照

■ログ： 2.6 を参照

■ウィンドウ： 2.7 を参照

■ヘルプ： 2.11, 2.14 を参照

ボタンパネルは、左から順に新規作成、開く、上書き保存、出力設定、コントロールパネル、artisoc ヘルプ、実行、ステップ実行、一時停止、停止、実行速度調整バー、モデル構築ガイドの表示となっています。

 新規作成：

新規ファイルが作成されます。

 開く：

既に存在するモデルファイルを開きます。

 上書き保存：

現在の設定を上書き保存します。

 出力設定：

出力設定を行います。詳しくは 2.2.3 をご参照ください。

 コントロールパネル：

コントロールパネルの設定を行います。詳しくは 2.2.5 をご参照ください。

 artisoc ヘルプ：

ヘルプを開きます。詳しくは 2.11 をご参照ください。

 実行、ステップ実行、一時停止、停止：

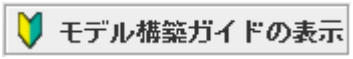
実行、ステップ実行、一時停止、停止については 2.3 をご参照ください。

 エージェント情報画面表示切替：

エージェント情報画面表示を切り替えます。

ステップ実行遅れ時間調整バー：

シミュレーションのステップの実行遅れ時間をスライドバーで調整できます。



モデル構築ガイドの表示：

モデル構築ガイドを表示します。

メニューボタンのグリッド部分（網掛け部分）をドラッグ＆ドロップすることにより、ボタンパネルを画面から切り離すことができます。

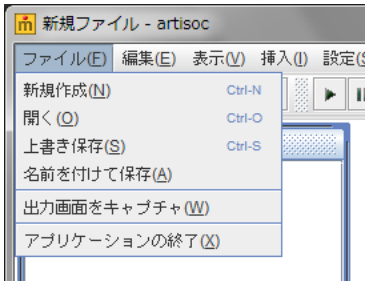
2.3. モデルを設定する

ここでは、エージェントや変数などのコンポーネントを設定する、モデル設定機能について解説します。

2.3.1. モデルファイルの入出力

シミュレーションを新規に作成したり、既存のモデルファイルを読込んだり、作成したシミュレーションの設定を保存したりする場合、次の図のようにファイルメニューのコマンドを選択します。シミュレーションの設定データは「モデルファイル（*.model）」として保存されます。

モデルファイルの新規作成



新規にシミュレーションを設定する場合、「ファイル」メニューから「新規作成」を選択します。

モデルファイルを開く

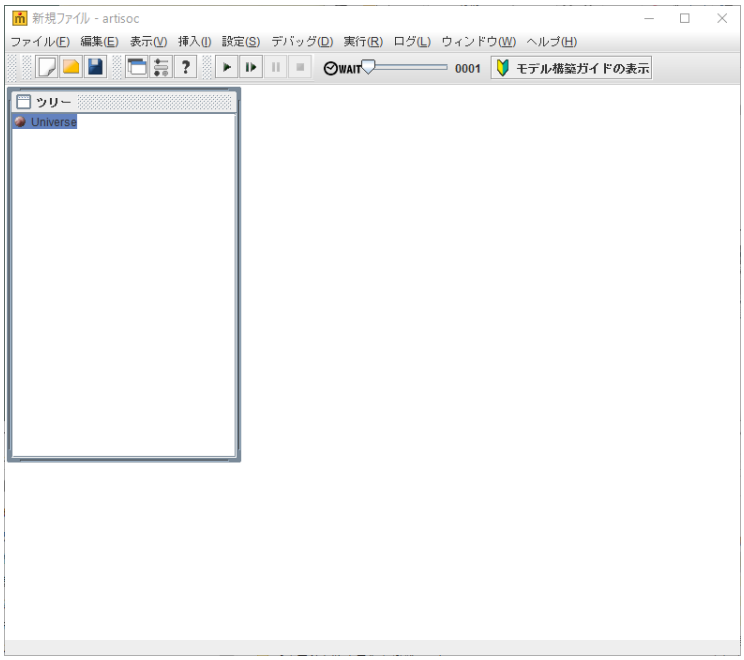
モデルファイルを開く場合、「ファイル」メニューから「開く」を選択します。

モデルファイルの保存

設定したモデルファイルを保存する場合、「ファイル」メニューから「上書き保存」または「名前をつけて保存」を選択します。

2.3.2. 新規作成

artisoc を起動もしくは新規作成を行うと、次の図のように画面が表示されます。



■コンポーネントツリー

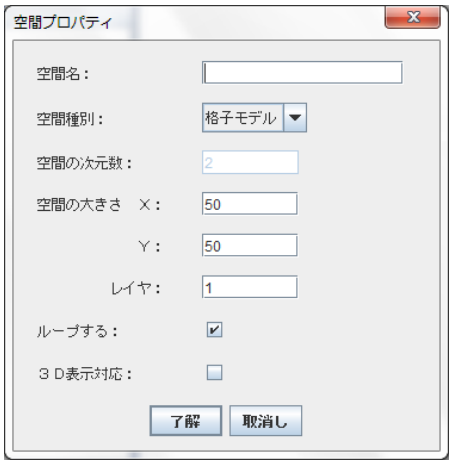
各コンポーネントを階層構造で表示する、このシミュレータの一番基本的な画面です。コンポーネントツリーの中には必ず一つの「Universe」が存在します。そしてその下に「空間」や「エージェント」、「変数」などのコンポーネントを配置して、シミュレーションを設定します。

2.3.3. 空間の新規作成

エージェントが平面上で移動する様を観察するために、空間を作成することが出来ます。

手順

1. **コンポーネントツリーの Universe を選択します。**
2. **メニューの「挿入」から「空間の追加」を選択します。**
3. **次のような「空間プロパティ」ダイアログが開きます。**



■空間名：

ユーザが任意の空間名を入力します。例えば、“空間”と入力します。

■空間種別：

空間の種類を選択します。格子モデルと六角形モデルのどちらかを選択します。

■空間の次元数：

空間ですので 2 次元の“2”が設定されています。

※3 次元表示対応の場合にも、空間は基本的に 2 次元空間となります。

■空間の大きさ：

エージェントや変数一つをひとマスとした空間の広さを設定します。X 軸（横方向）、Y 軸（縦方向）、レイヤー（階層）を指定します。初期値では 50×50×1 です。

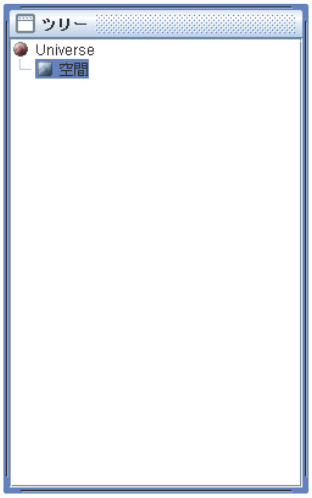
■端点の処理：

「ループする」にチェックを入れると、コンポーネントが空間の端まで来た時に反対側から現れます。「ループしない」場合は空間の端に留まります。

■３D 表示対応：

「3D マップ」として表示する空間の場合にはチェックを入れてください。チェックを入れた場合には、この空間以下に定義されたエージェントには「Z」変数と「Angle」変数が表示されます。ただし、チェックを入れた場合にも空間の扱いは基本的に 2 次元空間の場合と同じです。（現在の仕様では、３D 表示対応の場合でも組み込み関数は 2 次元空間としての処理を行います）

4.「了解」ボタンをクリックするとダイアログが閉じ、コンポーネントツリーに次の図のような空間が出現します。



こうして Universe の直下に空間を設定することが出来ました。

＜注意＞：空間は Universe の直下にしか作成することが出来ません。（概念的にわかりやすくするための仕様です）

2.3.4. エージェントの型の新規作成

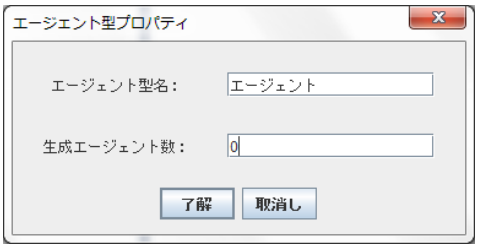
次にここでは、空間上にエージェント型を新規作成します。エージェント型は空間とは違って Universe の直下でも、他のエージェント型直下でも作成することが可能ですが、ここでは空間上で作成する場合を例としてあげておきます。

手順

1. エージェントを作成する空間（ここでは“空間”）を選択します。

2. 「挿入メニュー」の「エージェント型の追加」を選択します。

3. 次のような「エージェントプロパティ」ダイアログが開きます。



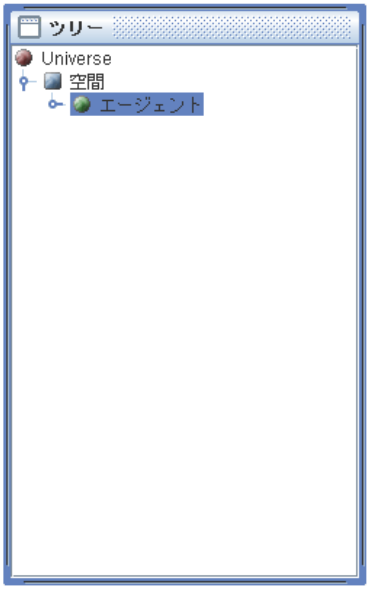
■エージェント型名：

任意のエージェント型名を入力します。例えば“エージェント”と入力します。

■生成エージェント数：

このエージェント型のエージェントを生成する数を入力します。ここに入力した数だけ、同じエージェント型のルールで動くエージェントが生成されます。

4. 「了解」ボタンをクリックするとダイアログが閉じ、コンポーネントツリーに次の図のようなエージェントが出現します。



こうして、空間の直下にエージェント型を新規作成することが出来ました。Universe の直下に新規作成する場合も、エージェント型の直下に子エージェント型を新規作成する場合も、同じ手順で作成します。

＜注＞：空間上にエージェント型を配置した場合は、「ID、X、Y、Layer　Direction」という名の５つの変数が自動的にエージェント型の下に配置されます。これらは、エージェントが座標情報を持つための変数です。

2.3.5.　エージェントルールの記述

エージェント型を作成してもルールを決定しないと生成したエージェントは何も仕事をしません。そのために各エージェント型に対してエージェントルールを記述する必要があります。

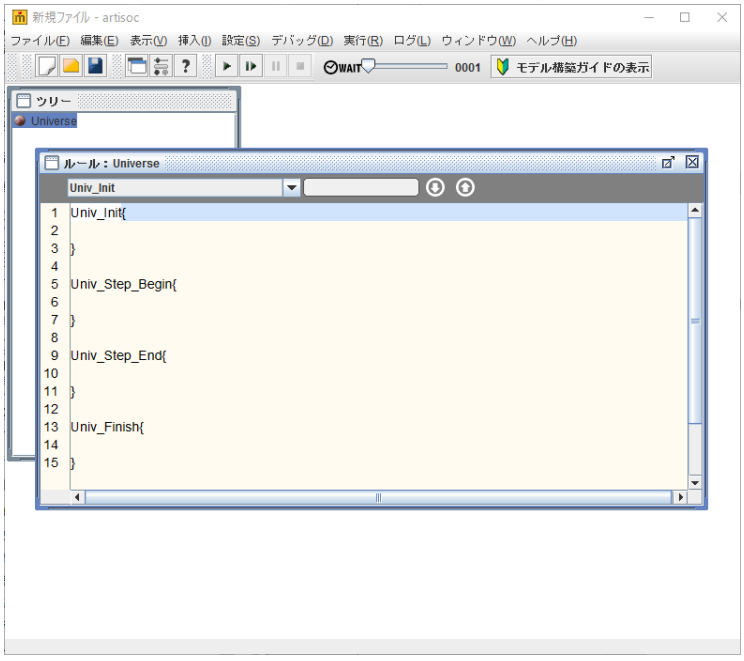
手順

1.　エージェント型を選択して「ルールウィンドウ」を開きます。

ルールウィンドウには、Universe へルールを書き込む「Universe ルールエディタ」と、エージェントへルールを書き込む「エージェントルールエディタ」の 2 種類があります。

ルールエディタを開くには、エージェントを選択して「表示メニュー」＞「ルールエディタ」を選択するか、エージェントを右クリックして、「ルールエディタ」を選択してください。

そうすると次のようなウィンドウが開きます。



2.　この中にエージェントルールを記述します。

“Agt_Init{　}”の“{”と“}”の間に、エージェントが生成される時に 1 回実行するルールを記述します。

“Agt_Step{　}”　の“{”と“}”の間に、シミュレーションのステップ毎に実行するルールを記述します。

Universe は特別に 4 種類のルールを記述することができます。

“Univ_Init{　}”の“{”と“}”の間に、シミュレーションが開始された最初に 1 回実行するルールを記述します。

“Univ_Step_Begin{　}”の“{”と“}”の間に、シミュレーションのステップの最初に実行するルールを記述します。

“Univ_Step_End{ }”の“{”と“}”の間に、シミュレーションのステップの最後に実行するルールを記述します。

“Univ_Finish{ }”の“{”と“}”の間に、シミュレーションが終了する直前に 1 回実行するルールを記述します。

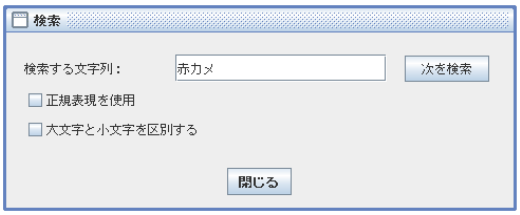
＜注意＞：既存のマルチエージェントシミュレータは「日本語環境が使えないこと」と「非常に難解なプログラミング環境を習得しなければシミュレーションを組めないこと」という 2 点の欠点を持っています。こうした点を踏まえてこのシミュレータは開発されました。とはいえ、ユーザが自由にシミュレーションを組みルールを設定するためには、プログラミング的なルールの記述は最低限必要になってしまいます。このルールに関しては、付属するサンプルモデルに記述されたルールなどを参考に研究してみてください。

今回 artisoc で使用されるルールの記述は、比較的習得が容易だとされるマイクロソフト社の“Visual Basic”の文法に準拠しています。必要な文法は「1.7.3 VisualBasic と artisoc との表記の違い」、「第 3 章エージェントルール文法」を参照してください。

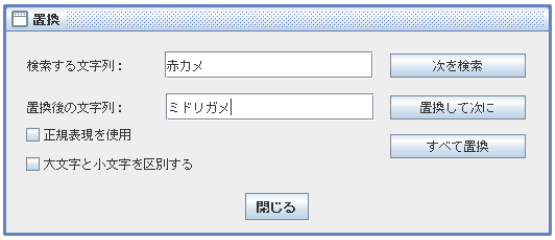
検索、置換機能

ルールエディタを開いている状況で「編集」メニューから「検索」「置換」を選択すると、検索、置換ウィンドウが開きます。

（1）検索画面

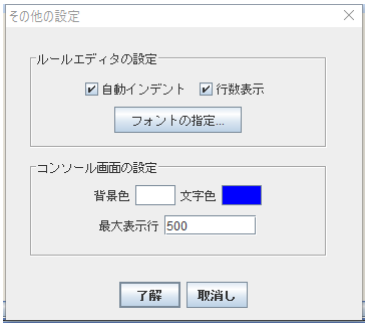


（2）置換画面



ルールエディタ・コンソール画面の設定

「設定」メニューの「その他の設定」を選択すると、ルールエディタの書体の設定をすることができます。



■自動インデント：

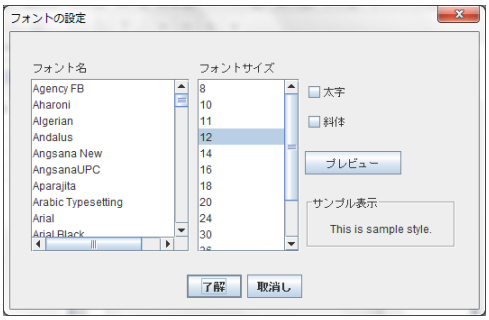
ここにチェックを入れると、ルールエディタでルールを記述する際に、自動的にインデント処理がされるようになります。

■色づけ：

ここにチェックを入れると、ルールエディタでルールを記述する際に、色づけ処理がされるようになります。このチェックをはずすとルールエディタのレスポンスが速くなります。

■フォントの指定：

「フォントの指定」ボタンをクリックすると、「フォントの設定」画面が表示されます。



■背景色：

コンソール画面の背景色を設定することができます。

■文字色：

コンソール画面の文字の色を設定することができます。

■表示最大行数：

コンソール画面に表示する文字の最大行数を設定します。指定した最大行数を超えた分は削除されます。初期値は 500 行です。

ここで設定したフォントやスタイル、サイズなどは、即座にルールエディタの表示に反映されます。

2.3.6. エージェント型の削除

削除したいエージェント型をクリックして選択して「編集」メニューから「削除」を選択するか、エージェント型を右クリックしてメニューから「削除」を選択します。

2.3.7. 変数の新規作成

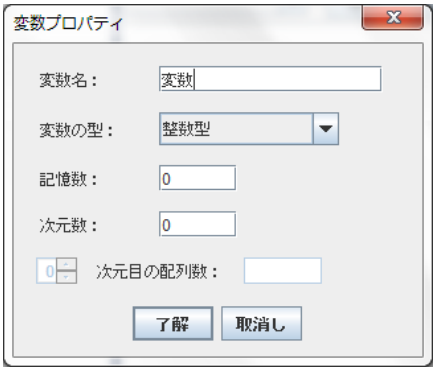
エージェントに変数を新規作成します。変数はエージェントの下なら何処にでも作成することが出来ます。

手順

1. 変数を追加するエージェント（ここでは“エージェント”）選択します。

2.「挿入メニュー」から「変数の追加」を選択します。

3. すると次のような「変数プロパティ」ダイアログが開きます。



■変数名：

任意の変数名を入力します。

■変数の型：

変数に入るデータに合わせて変数の型を選択する必要があります。種類は次の通りです。

型の種類	型の名前	値の範囲
ブール型	Boolean	真のとき True、偽のとき False の値を返す
文字列型	String	文字数は 0～（制限なし）
整数型	Integer	-2,147,483,648 ～　2,147,483,647 の整数
長整数型	Long	-9,223,372,036,854,775,808 ～9,223,372,036,854,775,807 の整数
実数型	Double	$-1.79769313486232 \times 10^{308}$ $\sim -4.94065645841247 \times 10^{-324}$ （負の場合） $4.94065645841247 \times 10^{-324}$ $\sim 1.79769313486232 \times 10^{308}$ （正の場合）
空間型	Space	モデルツリーに定義されている空間名 ※空間のサイズは縦幅 1～10,000、 横幅 1～10,000
エージェント種別型	AgtType	ツリーで定義されるエージェントの種別
エージェント型	Agt	エージェントそのもの。エージェントの実体値
エージェント集合型	AgtSet	エージェントの集合

基本的に文字には「文字列型」、整数には「整数型」、小数点を含む場合は「実数型」を選びましょう。「ブール型」は少し特殊な型で、True か False のどちらかの値が入ります。「エージェント型」はエージェントそのものを関数の引数に使いたい時に使うもので、「エージェント集合型」はそのエージェントの集合です。

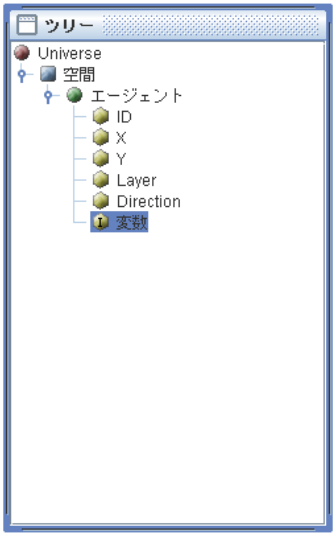
■記憶数：

artisoc では変数の過去の状態(値)を残しておくことができます。何ステップ分の状態を残すかを記憶数で指定します。過去の状態は GetHistory 関数で取得することが可能です。最大記憶数は 10,000 です。

■次元数：

変数はデータを入れる箱のようなものですが、原則として一つの変数には一つのデータしか入りません。新規作成時の変数は 0 次元（つまり平面状の点にあたる）になっており、データを一つ格納することになります。しかしそれでは同じ種類のたくさんのデータを扱う時に不便なので、次元数を増やすことで格納できるデータの数も増やすことが出来ます。ちなみに 1 次元の場合は 0 から「配列数」で設定した数（n）までの（n＋1）個のデータを格納できます。2 次元の場合は、1 次元目の配列数をｍ、2 次元目の配列数をｎとすると（m＋1）×（n＋1）個のデータを格納できます（ちなみにこれを「変数（m,n）」と表します)。配列数は 0 から始まるため個数に＋1 されることに気を付けましょう。次元数と配列数の最大数は 10,000 です。

4.「了解」ボタンをクリックするとダイアログが閉じ、ツリーに次の図のような変数が出現します。



＜注意＞：空間上にエージェント型を作成した時に、既に「ID」「X」「Y」「Layer」「Direction」という名の変数が作成されています。「ID」はエージェント種別毎にエージェントの１つずつを見分けるためにつけられる番号（0 以上）を表します。その他の変数は、空間上のエージェントの座標を表す変数で、空間が存在しない場合は作成されません。

こうして、エージェント型の下に変数“変数”を新規作成することが出来ました。Universe の直下に新規作成する場合も、空間の直下に新規作成する場合も、同じ手順で作成します。

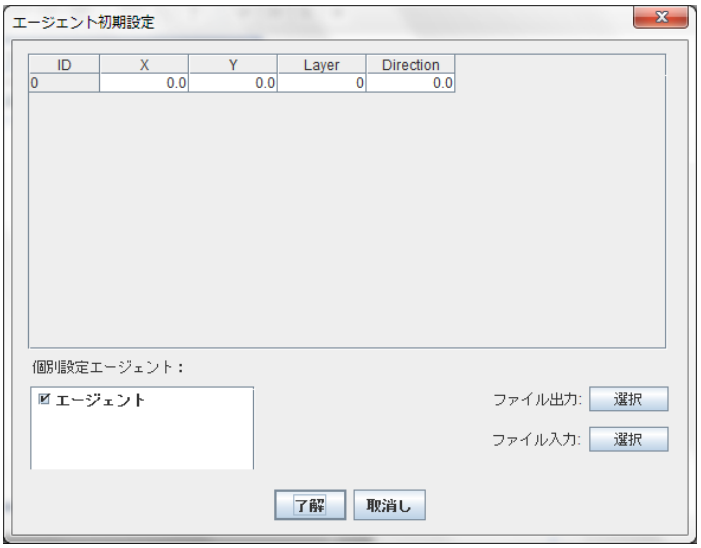
2.4. シミュレーション設定機能

2.4.1. 初期値設定ダイアログ

各コンポーネントの初期値を設定します。ツリー上の各コンポーネントを選択した上で、「設定」メニューから「初期値設定」を選択すると、初期値設定ダイアログが開きます。この初期値を設定する画面は選択したコンポーネントの種類によって変わります。

エージェント及び変数の初期値設定ダイアログ

コンポーネントツリー上の「エージェント」や「変数」を選択し、「設定」メニューから「初期値設定」を選択すると、このダイアログが表示されます。ここでは、「エージェント」や「変数」にシミュレーション開始時の初期値を設定することが出来ます。(「エージェント」の変数は、エージェント数が 0 の時は設定できません)



※エージェントの数が 1 のとき

■個別設定エージェント：

エージェントの個数を複数に設定している場合、ここにチェックを入れておけば、各々のエージェントに対して個別に数値を設定することが出来ます。

■ファイル入力（出力）：

このボタンをクリックすると、初期値のデータを既存ファイルから読込んだり、ファイルに保存したりすることが出来ます。

2.4.2. 空間用描画ダイアログ

空間上のエージェント配置やネットワークの設定、空間上の変数設定を描画ダイアログを用いて簡単に行うことができます。但し、描画ダイアログでは設定のみを行うため、画面出力とは別になります。

ツリー上の空間コンポーネントを選択した上で、「設定」メニューから「初期値設定」を選ぶと、描画ダイアログが開きます。

使用前の準備

描画ダイアログを用いてエージェントの配置を行う場合はポイントエージェント、リンクの接続を行う場合はリンクエージェントがツリー上に定義されている必要があります

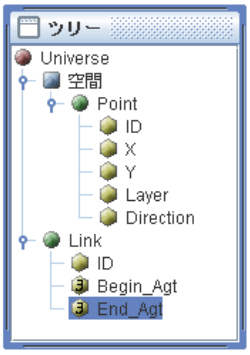
■ポイントエージェント：

マップ上に配置されるエージェントです。空間直下に定義されている全エージェントはポイントエージェントとして認識されます。

■リンクエージェント：

リンクエージェントはポイントエージェント同士の接続情報を格納します。

リンクエージェントは Universe の直下に作成し、2 つのエージェント型変数（Begin_Agt と End_Agt）を持たせておく必要があります。リンクエージェントの接続設定が行われると、起点と終点になるポイントエージェントの UniqueID がエージェント型変数に格納されます。リンクエージェントには Begin_Agt と End_Agt 以外の変数を追加することが可能なので、付加情報（道路の幅や傾斜など）を持たせることもできます。

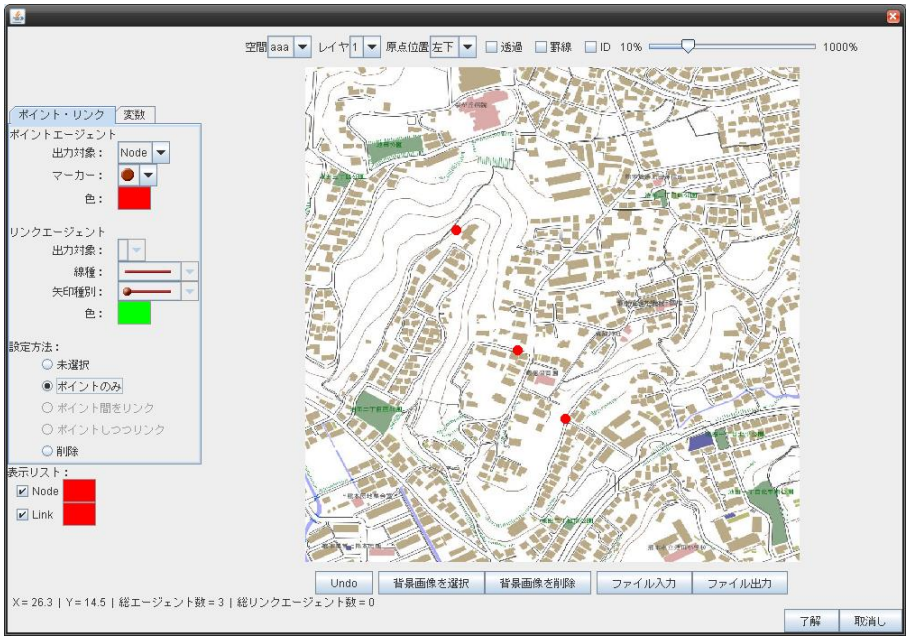


● ポイントエージェント、リンクエージェント名は任意で与えられる

● リンクエージェントに「Begin_Agt」「End_Agt」をエージェント型変数として定義しない場合、描画ダイアログで表示されない

ポイントエージェント・リンクエージェント設定

ポイントエージェント及びリンクエージェントの配置は「ポイント・リンク」タブで行います。



■タブ：

タブを選択して、設定対象を切り替えます。標準は「ポイント・リンク」タブ。

■空間：

配置を行っている空間名です。プルダウンで他の空間に切り替えることができます。

■レイヤ：

配置を行っている空間のレイヤ番号です。プルダウンで他のレイヤに切り替えることができます。

■透過：

チェックを入れると現在設定中のレイヤ以外に配置されたエージェントが表示されます。

■罫線：

チェックを入れると空間のセルに合わせて罫線が表示されます

■ID：

チェックを入れるとエージェントの ID が表示されます。

■倍率：

スライドバーを動かすと表示画面の倍率が変化します。

■空間表示：

設定中の空間が表示されます。画面内でドラッグを行うと画面内の表示がマウスの動きに合わせて移動します。

■ポイントエージェント：

配置を行うポイントエージェントの種類及び表示画面内でのマーカーの種類と色を指定します。

■リンクエージェント：

ポイントエージェント接続に用いるリンクエージェントの種類及び表示画面内での線種、矢印種別、色を指定します。

■設定方法：

表示画面内で行う作業を指定します。設定方法は「未選択」「ポイントのみ」「ポイント間をリンク」「ポイントしつつリンク」「削除」の 5 つがあります。

【未選択】

ポイントエージェントの配置やリンクエージェントの接続を行いません。

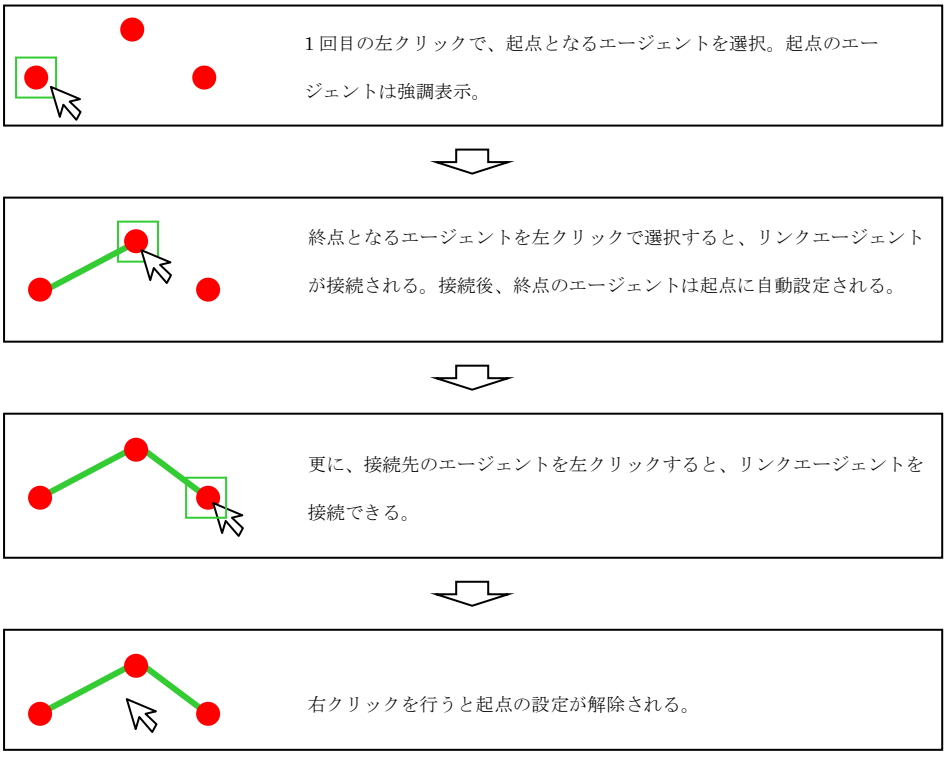
【ポイントのみ】

表示画面内で左クリックを行うと、クリックされた場所に選択中のポイントエージェントが 1 つ配置されます

【ポイント間をリンク】

選択中のリンクエージェントを用いてポイントエージェントを接続するようになります。

リンクエージェントの接続



【ポイントしつつリンク】

「ポイント間をリンク」と同様に、選択中のリンクエージェントを用いてポイントエージェントを接続するようになります。「ポイントしつつリンク」ではポイントエージェントが存在しない場所を左クリックした場合、ポイントエージェントを配置するので、ポイントエージェントの配置とリンクエージェントの接続が同時に行えます。

【削除】

表示画面内でエージェントを左クリックすると削除されるようになります。リンクエージェントが接続されているポイントエージェントを削除すると、リンクエージェントも同時に削除されます。

■表示リスト：

チェックの入った項目のみ、画面内に表示されます。

X，Y，総エージェント数，総リンクエージェント数：

X、Y にはカーソルの座標が表示されます。総エージェント数、総リンクエージェント数には設定中のポイントエージェントの総数とリンクエージェントの総数が表示されます。

■Undo：

1 つ前の動作を実行する前に戻ります。

■背景画像を選択：

画面に背景画像を表示することができます。ファイル選択のダイアログが出てくるので、背景に表示させたい画像を選択してください。

■背景画像を削除：

背景画像を削除します。

■ファイル入力：

ファイルからポイントエージェント、リンクエージェント、変数のデータを読み込みます。ファイルは以下のフォーマットに従ったものしか使用できません。

【ファイルフォーマット】

（例）

Point,Point_Agt 型名,ID,X 座標,Y 座標,レイヤ番号

Point, Point_Agt 型名,ID,X 座標,Y 座標,レイヤ番号

Variable,空間変数名,空間の幅,空間の高さ

Layer,レイヤ番号

変数値(1,1), 変数値(1,2),..., 変数値(1,x)

.

.

変数値(y,1), 変数値(y,2),..., 変数値(y,x)

Layer,レイヤ番号

変数値(1,1), 変数値(1,2),..., 変数値(1,x)

.

.

変数値(y,1), 変数値(y,2),..., 変数値(y,x)

.

.

.

Link,Link_Agt 型名,ID,起点 Point_Agt 型名,起点 Point_Agt の ID, 終点 Point_Agt 型名, 終点 Point_Agt の ID

Link,Link_Agt 型名,ID,起点 Point_Agt 型名,起点 Point_Agt の ID, 終点 Point_Agt 型名, 終点 Point_Agt の ID

＊ポイントエージェント＝Point_Agt

＊リンクエージェント＝Link_Agt

■ファイル出力：

ポイントエージェント、リンクエージェント、変数の全データをファイルに出力します。ファイルはファイル入力で述べたフォーマットに従って出力されます。

空間変数の設定

空間変数の設定は「変数」タブで行います。

■変数:

設定対象、マーカーの種類、最小値・最大値の色を設定できます。

■設定値：

空間変数に設定する値を入力してください

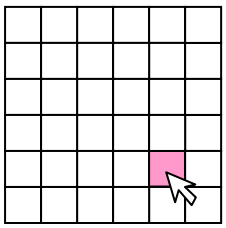
■指定方法：

表示画面内で行う作業を指定します。指定方法は「未選択」「クリック」「矩形」「直線」の 4 つがあります。

【未選択】

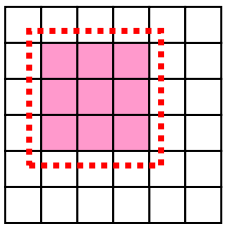
ポイントエージェントの配置やリンクエージェントの接続を行いません。

【クリック】



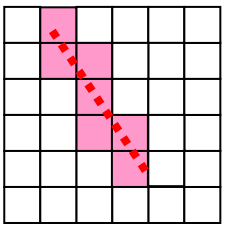
クリックしたセルの空間変数に設定値の値が入力される

【矩形】



範囲指定した中に入るセルの空間変数に設定値の値が入力される

【直線】



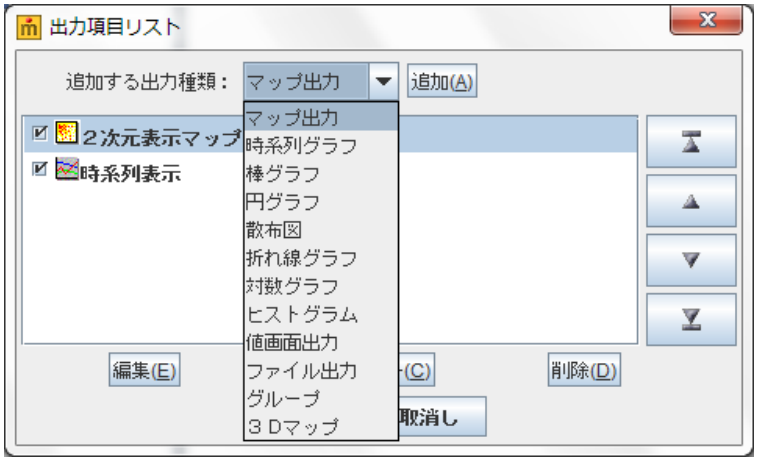
直線を引き、直線が含まれているセルの空間変数に設定値が入力される

変数タブ内以外は、ポイントエージェント配置及びリンクエージェント接続の時と同じです。

2.4.3. 出力設定ダイアログ

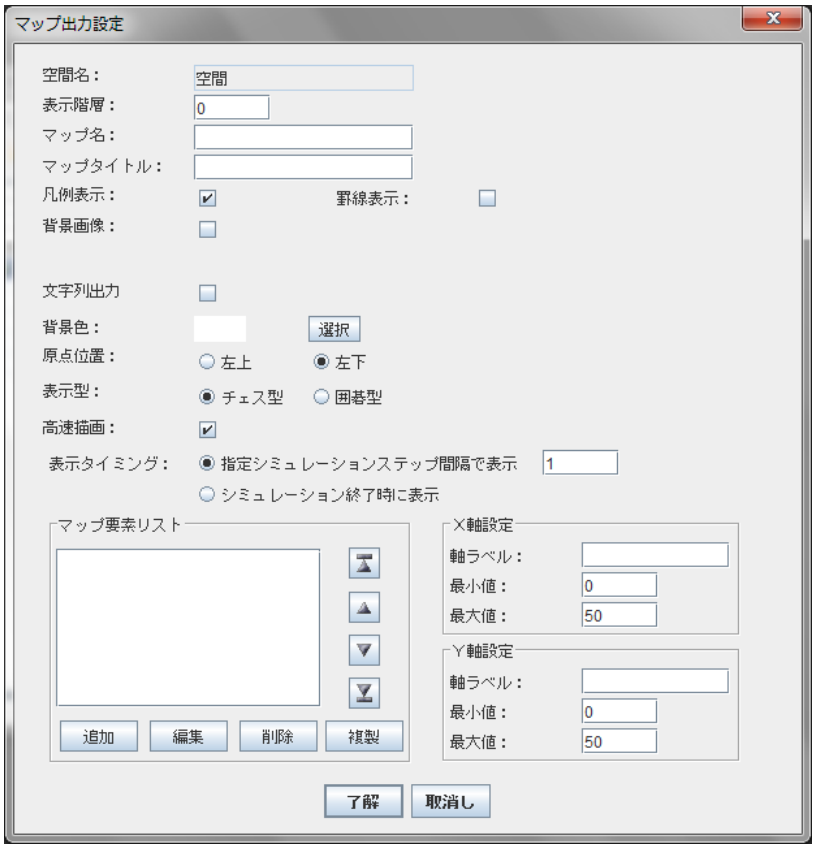
「設定」メニューの「出力設定」を選ぶと次の「出力項目リスト」ダイアログが表示されます。このダイアログの「追加する出力種類」から各種グラフやファイルを選択し、「追加」ボタンをクリックすることによってグラフやファイルの出力方法を設定することが出来ます。複数の出力が必要な場合、出力方法を追加する必要があります。右側の移動アイコンにより、出力順序を変更することが出来ます。表示チェックボックスにより、表示／非表示を選択することが出来ます。

出力されたグラフは、マウス操作によって拡大することができます。拡大したい部分を左上から右下に向かって選択することで拡大表示することができます。一度拡大表示したグラフを元に戻したい時は、グラフのいずれかの部分を右下から左上に向けて選択してください。



2.4.3.1.　二次元表示マップ出力設定

二次元の表示マップ上にエージェントや変数を配置し、それらの移動や増減を見る出力方法です。「追加する出力種類」で「マップ出力」を選択すると、次のダイアログが表示されます。



■空間名：

表示対象の空間名が表示されます。

■表示階層：

表示する階層を指定します。空間設定で「空間の大きさ-レイヤ」の値が 1 のときは階層 0、1 以上を設定した場合は表示したい階層（＝レイヤ数）を指定できます。

■マップ名：

このマップを呼び出す時に参照する名称です。

■マップタイトル：

出力画面にて表示されるマップにこの「マップタイトル」が表示されます。

■凡例表示：

ここにチェックを入れておくと、マップの横に凡例が表示されます。

■罫線表示：

ここにチェックを入れておくと、表示される二次元マップに縦横の罫線が描画されます。

■背景画像：

ここにチェックを入れると、表示される二次元マップの背景画像を指定することができます。
背景画像として「ファイル名」（モデルファイルが存在するフォルダからの相対パスで指定）、もしくは「ファイル指定変数名」（Universe 直下の文字列型変数がリスト表示）を選択できます。

■文字列出力：

ここにチェックを入れた場合、Universe 以下の文字列変数を指定できます。表示される二次元マップに指定された文字列変数の値が表示されます。

■背景色：

ここで指定した色が、表示される二次元マップの背景色として設定されます。

■原点位置：

原点の位置をコンピュータ等で一般的な左上にするか、数学等で一般的な左下にするか、選択することができます。

■表示型：

エージェントが罫線で囲まれるマスの中に表示されるチェス型か、エージェントが罫線の交点に表示される囲碁型か、選択することができます。

■高速描画：
各エージェントの表示を簡略化することで、artisoc の処理速度を高速化します。エージェントの数が多く、計算に時間がかかるときに使用してください。

■表示タイミング：
出力画面の更新タイミングを設定できます。
一定ステップ毎に更新させたい時は「指定シミュレーション間隔で表示」にチェックを入れてください。初期状態では 1 が入力されており、1 ステップ実行される度に 1 回マップ出力画面が更新されます。
「シミュレーション終了時」にチェックを入れるとシミュレーション終了時に表示されます。

■X 軸設定　Y 軸設定：

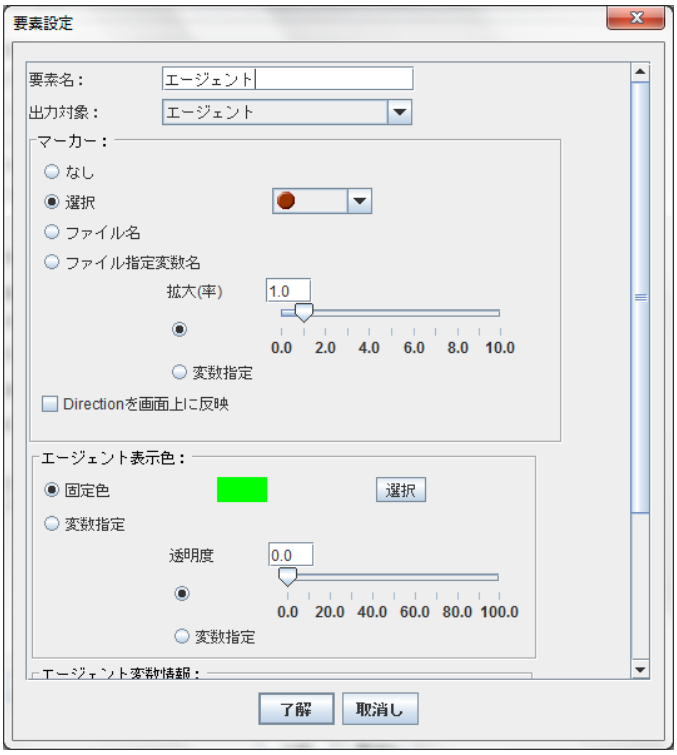
最小値と最大値を設定することによって、マップ上のある特定の部分だけを表示することが可能です。(初期値では空間の広さの値が最大値になっています)

■マップ要素リスト：

マップ上に表示するエージェントや変数をマップ要素として登録します。
もしも、画面上に複数の要素を同時に表示したい場合は、それら要素毎に（ex.エージェント 1、エージェント 2、変数 1、etc.）マップ要素として追加する必要があります。(つまり要素の数だけ追加リストに並ぶことになります)
「編集」ボタンをクリックすると、任意の既存のマップ要素を編集することが出来ます。また「削除」ボタンによって、任意の既存のマップ要素を削除することが出来ます。

■マップ要素リストの追加

マップ要素リストの「追加」ボタンをクリックすると、次の画面が表示されます。設定項目が多いため、画面右側にスクロールバーを追加しています。必要に応じてスクロールしてください。



■要素名：

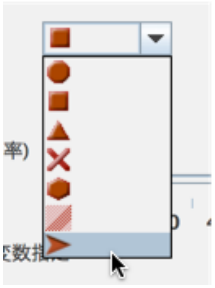
このマップ要素に任意の名称をつけます。

■出力対象：

そのマップ上での動きを見たい要素（エージェントまたは空間に定義された変数）を選択します。

■マーカー：

マップ上の要素にマーカーをどのように表示するかどうかを選択します。「なし」ではマーカーは表示されません。「選択」では「●■▲×（六角刑）（網掛け）（矢印）」の中から選択します。「ファイル名」では任意の場所に置かれた画像ファイル（現在はアイコンファイルのみ対応）へのパス名を入力することによりマーカーを画像に置き換えることができます。「ファイル指定変数名」では、Universe 上にある変数を選択します。



エージェントの表示サイズを「拡大（率）」で指定します。拡大倍率が１のときには、１セルのサイズと同じサイズでマーカーが表示されます。倍率をテキストボックスで直接設定するか、スクロールバーで設定してください。また、拡大（率）は変数経由で指定することも可能です。拡大率のラジオボタンを「変数指定」にして、拡大率を指定する変数名をプルダウンから選択します。

「Direction を画面上に反映」にチェックを入れると、エージェントの Direction 変数の値に従ってマーカーの向きが変化するようになります。

■エージェント表示色：(エージェントが出力対象の場合)

任意の一色を指定する「固定色」と指定した変数の値によって動的に色が変化する「変数指定」のどちらかを選択できます。なお、デフォルト色は登録順に、赤→青→緑→紫→水色→黄→灰色が表示されます。

エージェントの表示透明度を「透明度」で指定します。透明度は０から１００まで指定可能で、値が大きいほど透明となります。透明度をテキストボックスで直接設定するか、スクロールバーで設定してください。また、透明度は変数経由で指定することも可能です。透明度のラジオボタンを「変数指定」にして、透明度を指定する変数名をプルダウンから選択します。

■表示色：(空間の変数が出力対象の場合)

空間の変数を出力対象とした場合、出力対象の変数の値によって色が表示されます。

ここでは色の表示のされ方を指定します。「グラデーション指定」と「変数の値を RGB の値とする」の２つの指定の仕方があります。「グラデーション指定」の場合には、指定された変数の範囲に該当する色が表示されます。表示する色に対応する最小の値と色、最大の値と色を指定します。この間にある値に対応する色は、最小値に近いほど、最小値に設定した色に近づき、最大値に近くなるほど最大値に設定した色に近づきます。

変数の表示透明度を「透明度」で指定します。透明度は０から１００まで指定可能で、値が大きいほど透明となります。透明度をテキストボックスで直接設定するか、スクロールバーで設定してください。また、透明度は変数経由で指定することも可能です。透明度のラジオボタンを「変数指定」にして、透明度を指定する変数名をプルダウンから選択します。

■エージェント変数情報：（エージェントが出力対象の場合）

「情報表示」にチェックを入れると、マップ上のエージェントの上に指定された変数の数値が表示されます。

■線を引く：（エージェントが出力対象の場合）

出力対象のエージェントがエージェント集合変数を持つ場合は、出力対象エージェントとエージェント集合の要素それぞれとを線で結ぶことができます。太さや端点の種類を選択することもできます。線を引くにチェックを入れると、線引き対象、線種、矢印種別、色の設定を選択できるようになります。

■線引き対象：

各エージェントが持つ、線を引きたいエージェント集合変数を指定します。

■線種：

線の種類を実線、点線、一点破線の中から指定します。変数指定をしない場合は、左のタブの中から選択してください。変数指定を行う場合の対応は、実践＝１、点線＝２、一点破線＝３　です。

■矢印種別：

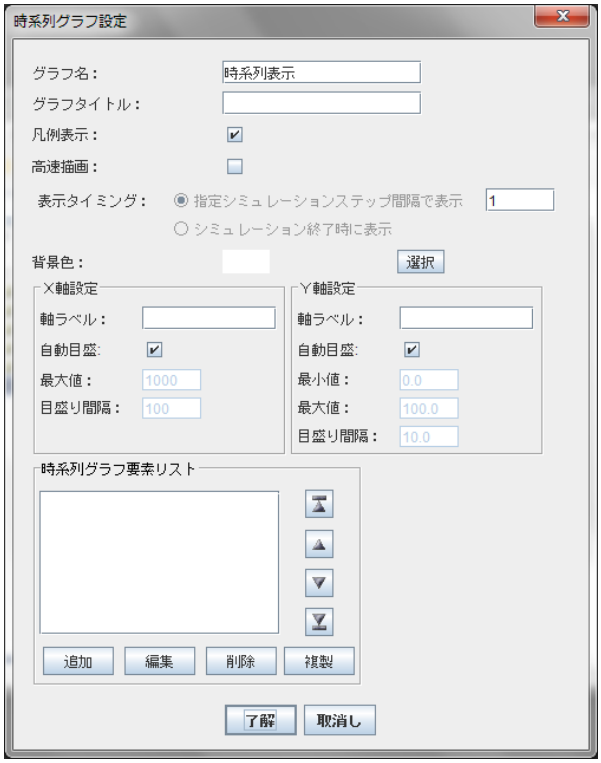
矢印の種別を矢印無し、相手から自分方向、自分から相手方向、双方向の中から指定します。変数指定をしない場合は、左のタブの中から選択してください。変数指定を行う場合の対応は、矢印無し＝１、相手から自分方向＝２、自分から相手方向＝３、双方向＝４です

■色の設定：

線の色を設定することができます。変数指定しない場合は左の選択ボタンから指定してください。変数指定を行う場合は、エージェントに整数型の変数を加え、選択してください。

2.4.3.2.　時系列グラフ出力設定

コンポーネントを時系列の折れ線グラフで表示する出力方法です。「追加する出力種類」で「時系列グラフ」を選択すると、次のダイアログが表示されます。



■グラフ名：

このグラフを呼び出す時に参照する名称です。

■グラフタイトル：

出力画面にて表示されるグラフにこの「グラフタイトル」が表示されます。

■凡例表示：

ここにチェックを入れておくと、グラフの横に凡例が表示されます。

■高速描画：

グラフ表示を簡略化することで、artisoc の処理速度を高速化します。計算に時間がかかるときに使用してください。

■表示タイミング：

出力画面の更新タイミングを設定できます。

一定ステップ毎に更新させたい時は「指定シミュレーション間隔で表示」にチェックを入れてください。初期状態では 1 が入力さ

れており、１ステップ実行される度に１回マップ出力画面が更新されます。

「シミュレーション終了時」にチェックを入れるとシミュレーション終了時に表示されます。

■軸ラベル：

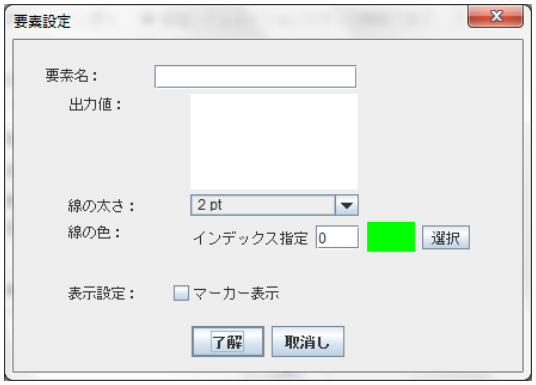
グラフを表示する時にその軸のラベルとしてグラフとともに表示されます。

■自動設定：

自動設定にチェックを入れることで、表示されるグラフの最小値、最大値、目盛の間隔を自動的に変化させます。自動設定を解除する場合は最小値、最大値、目盛を入力し、指定してください。

■グラフ要素リスト：

「追加」ボタンをクリックすると次の画面が表示されます。



■要素名：

このグラフ要素に任意の名称をつけます。

■出力値：

そのグラフの経緯を見たい要素（変数等）を入力します。

■線の太さ：　■線の色：

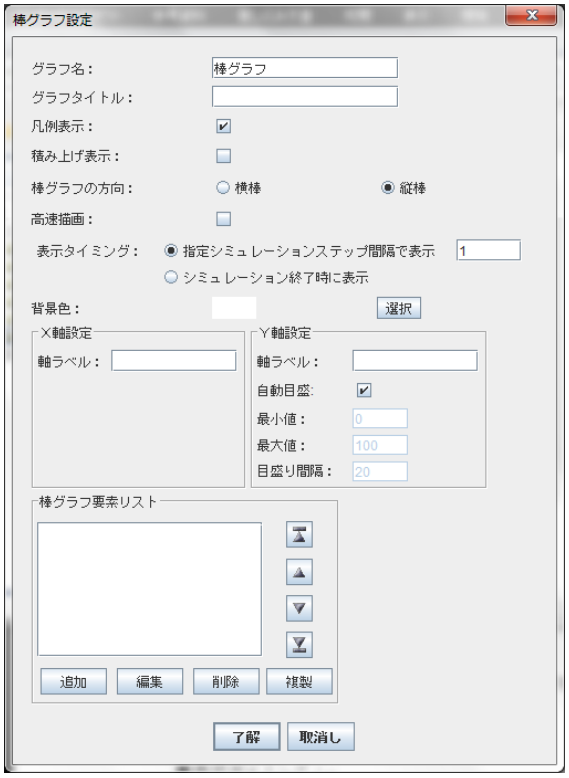
グラフの太さと色を設定します。なお、デフォルト色は登録順に、赤→青→緑→紫→水色→黄→灰色が表示されます。

■マーカー表示：

ここにチェックを入れるとグラフ上にマーカーが表示されます。

2.4.3.3.　棒グラフ出力設定

コンポーネントを棒グラフで表示する出力方法です。「追加する出力種類」で「棒グラフ」を選択すると、次のダイアログが表示されます。



■グラフ名：

このグラフを呼び出す時に参照する名称です。

■グラフタイトル：

出力画面にて表示されるグラフにこの「グラフタイトル」が表示されます。

■凡例表示：

ここにチェックを入れておくと、グラフの横に凡例が表示されます。

■積み上げ表示：

ここにチェックを入れておくと、設定された要素が積み上げられた棒グラフが表示されます。

（積み上げ表示の場合は、棒グラフの棒は１本となります）

■高速描画：

グラフ表示を簡略化することで、artisoc の処理速度を高速化します。計算に時間がかかるときに使用してください。

■表示タイミング：

出力画面の更新タイミングを設定できます。

一定ステップ毎に更新させたい時は「指定シミュレーション間隔で表示」にチェックを入れてください。初期状態では１が入力されており、１ステップ実行される度に１回マップ出力画面が更新されます。

「シミュレーション終了時」にチェックを入れるとシミュレーション終了時に表示されます。

■背景色：

グラフの背景の色を変更することができます。

■軸ラベル：

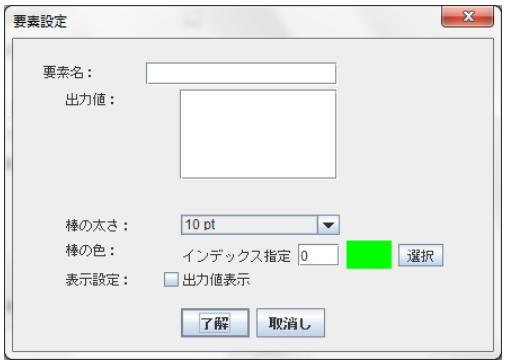
グラフを表示する時にその軸のラベルとしてグラフとともに表示されます。

■自動設定：

自動設定にチェックを入れることで、表示されるグラフの最小値、最大値、目盛の間隔を自動的に変化させます。自動設定を解除する場合は最小値、最大値、目盛を入力し、指定してください。

■グラフ要素リスト：

「追加」ボタンをクリックすると次の画面が表示されます。



■要素名：

このグラフ要素に任意の名称をつけます。

■出力値：

そのグラフの経緯を見たい要素（変数等）を入力します。

■棒の太さ：　■棒の色：

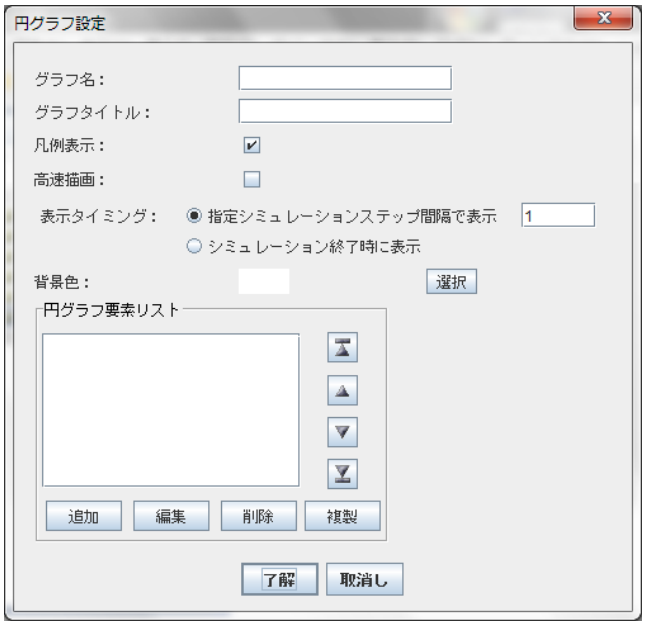
グラフの太さと色を設定します。なお、デフォルト色は登録順に、赤→青→緑→紫→水色→黄→灰色が表示されます。

■表示設定：

出力値表示のボックスにチェックを入れることで、棒グラフの手前に変数の値を表示するようにします。

2.4.3.4.　円グラフ出力設定

コンポーネントを棒グラフで表示する出力方法です。「追加する出力種類」で「円グラフ」を選択すると、次のダイアログが表示されます。



■グラフ名：

このグラフを呼び出す時に参照する名称です。

■グラフタイトル：

出力画面にて表示されるグラフにこの「グラフタイトル」が表示されます。

■凡例表示：

ここにチェックを入れておくと、グラフの横に凡例が表示されます。

■高速描画：

グラフ表示を簡略化することで、artisoc の処理速度を高速化します。計算に時間がかかるときに使用してください。

■表示タイミング：

出力画面の更新タイミングを設定できます。

一定ステップ毎に更新させたい時は「指定シミュレーション間隔で表示」にチェックを入れてください。初期状態では 1 が入力されており、1 ステップ実行される度に 1 回マップ出力画面が更新されます。

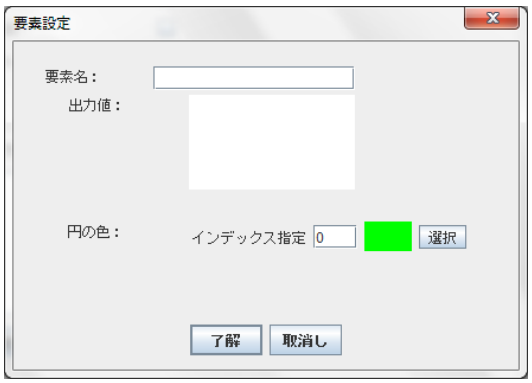
「シミュレーション終了時」にチェックを入れるとシミュレーション終了時に表示されます。

■背景色：

背景の色を変更することができます。

■グラフ要素リスト：

「追加」ボタンをクリックすると次の画面が表示されます。



■要素名：

このグラフ要素に任意の名称をつけます。

■出力値：

そのグラフの経緯を見たい要素（変数等）を入力します。

■円の色：

その変数を表す領域の色を変更することができます。

※補足

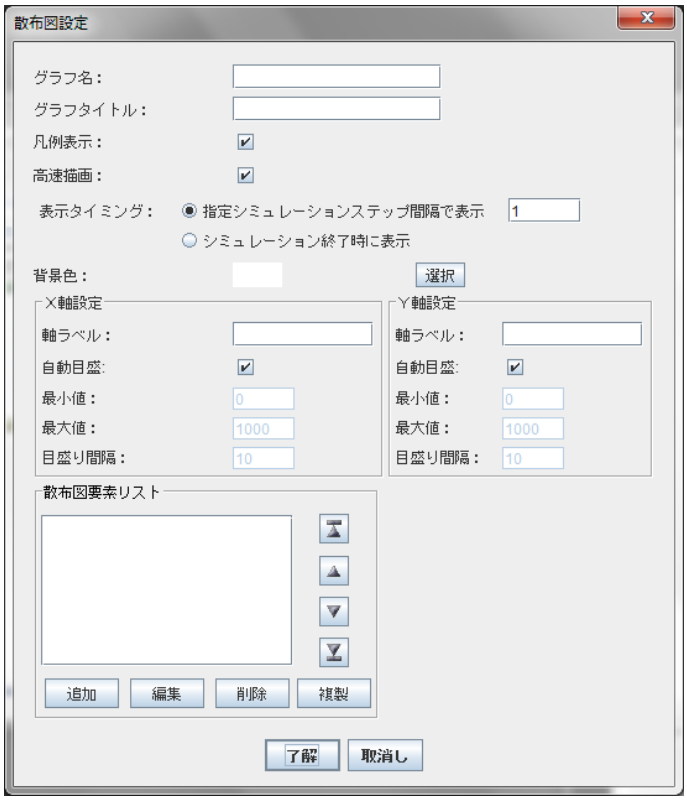
円グラフで表示する出力値に配列変数を指定することで、配列の各値を要素とした円グラフが表示されます。

例えば、Universe 以下に Data(5)： Integer という配列変数を定義し、Data(0)～Data(4)の各要素に値を格納したあと、円グラフの要素設定の出力値で「Universe.Data」と指定することで Data(0)～Data(4)の 5 つの要素の円グラフが表示されます。

配列変数のインデックスごとに色を指定することができます。設定する色の対象の配列インデックスを「インデックス指定」でインデックス番号を指定し、色を指定します。

2.4.3.5. 散布図出力設定

コンポーネントを棒グラフで表示する出力方法です。「追加する出力種類」で「散布図」を選択すると、次のダイアログが表示されます。



■グラフ名：

このグラフを呼び出す時に参照する名称です。

■グラフタイトル：

出力画面にて表示されるグラフにこの「グラフタイトル」が表示されます。

■凡例表示：

ここにチェックを入れておくと、グラフの横に凡例が表示されます。

■高速描画：

グラフ表示を簡略化することで、artisoc の処理速度を高速化します。計算に時間がかかるときに使用してください。

■表示タイミング：

出力画面の更新タイミングを設定できます。

一定ステップ毎に更新させたい時は「指定シミュレーション間隔で表示」にチェックを入れてください。初期状態では１が入力されており、１ステップ実行される度に１回マップ出力画面が更新されます。

「シミュレーション終了時」にチェックを入れるとシミュレーション終了時に表示されます。

■背景色：

背景の色を変更することができます。

■軸ラベル：

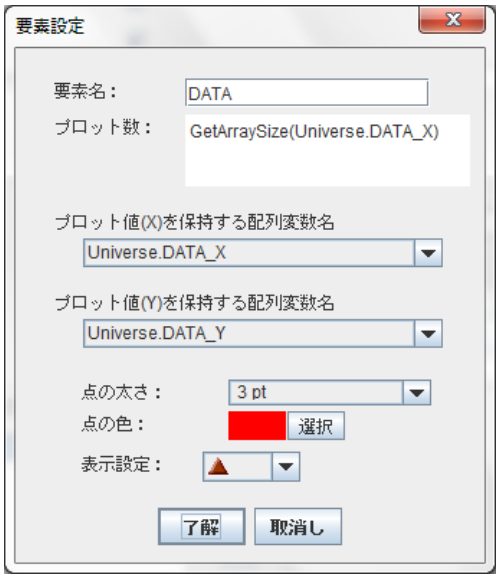
グラフを表示する時にその軸のラベルとしてグラフとともに表示されます。

■自動設定：

自動設定にチェックを入れることで、表示されるグラフの最小値、最大値、目盛の間隔を自動的に変化させます。自動設定を解除する場合は最小値、最大値、目盛を入力し、指定してください。

■グラフ要素リスト：

「追加」ボタンをクリックすると次の画面が表示されます。



■要素名：

このグラフ要素に任意の名称をつけます。

■プロット数：

表示させたいプロットの数を入力してください。指定された配列変数のうち、配列変数のインデックスのうち、０からここで指定した値までの値が描画されます。なお、ルール同様に変数や関数で指定することが可能です。（※参考：GetArraySize(【配列変数名】)で、引数に指定した配列変数のサイズを取得可能です）

■プロット値(X)を保持する配列変数名：プロット値(Y)を保持する配列変数名：

表示させるプロットの X の値と Y の値を決める変数を選択してください。ここには、配列変数を指定することで、上記指定されたプロット数だけの対象配列変数内の値をプロットします。

■点の太さ：

点の大きさを変更することができます。

■点の色：

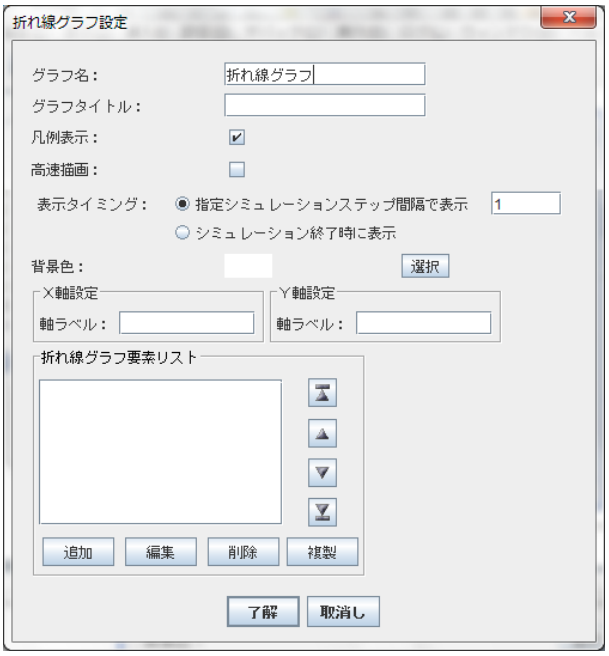
点の色を変更することができます。

■表示設定：

プロットする形状を設定することができます。

2.4.3.6. 折れ線グラフ出力設定

コンポーネントを折れ線グラフで表示する出力方法です。「追加する出力種類」で「折れ線グラフ」を選択すると、次のダイアログが表示されます。



■グラフ名：

このグラフを呼び出す時に参照する名称です。

■グラフタイトル：

出力画面にて表示されるグラフにこの「グラフタイトル」が表示されます。

■凡例表示：

ここにチェックを入れておくと、グラフの横に凡例が表示されます。

■高速描画：

グラフ表示を簡略化することで、artisoc の処理速度を高速化します。計算に時間がかかるときに使用してください。

■表示タイミング：

出力画面の更新タイミングを設定できます。

一定ステップ毎に更新させたい時は「指定シミュレーション間隔で表示」にチェックを入れてください。初期状態では 1 が入力されており、1 ステップ実行される度に 1 回マップ出力画面が更新されます。

「シミュレーション終了時」にチェックを入れるとシミュレーション終了時に表示されます。

■背景色：

背景の色を変更することができます。

■軸ラベル：

グラフを表示する時にその軸のラベルとしてグラフとともに表示されます。

■グラフ要素リスト：

「追加」ボタンをクリックすると次の画面が表示されます。

■要素名：

このグラフ要素に任意の名称をつけます。

■プロット数：

表示させたいプロットの数を入力してください。指定された配列変数のうち、配列変数のインデックスのうち、0からここで指定した値までの値が描画されます。なお、ルール同様に変数や関数で指定することが可能です。(※参考：GetArraySize(【配列変数名】)で、引数に指定した配列変数のサイズを取得可能です)

■プロット値(X)を保持する配列変数名：プロット値(Y)を保持する配列変数名：

表示させるプロットのXの値とYの値を決める変数を選択してください。配列の変数を指定することで、複数のプロットを行うことができます。

■線の太さ：

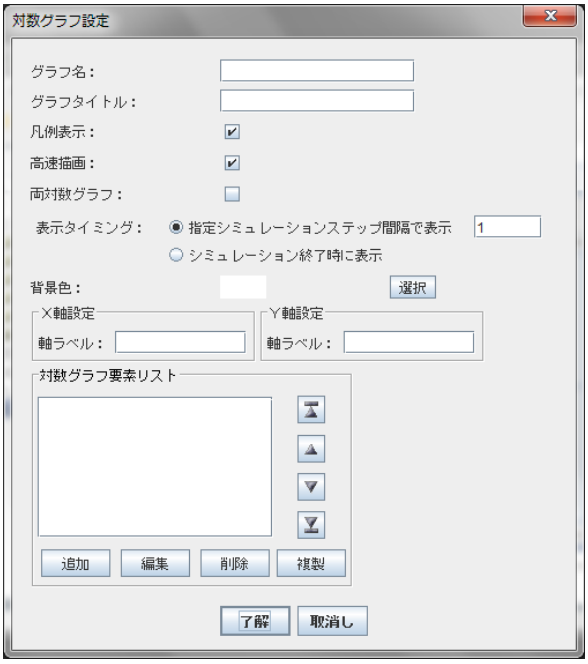
線の太さを変更することができます。

■線の色：

線の色を変更することができます。

2.4.3.7. 対数グラフ出力設定

コンポーネントを対数グラフで表示する出力方法です。「追加する出力種類」で「対数グラフ」を選択すると、次のダイアログが表示されます。



■グラフ名：

このグラフを呼び出す時に参照する名称です。

■グラフタイトル：

出力画面にて表示されるグラフにこの「グラフタイトル」が表示されます。

■凡例表示：

ここにチェックを入れておくと、グラフの横に凡例が表示されます。

■高速描画：

グラフ表示を簡略化することで、artisoc の処理速度を高速化します。計算に時間がかかるときに使用してください。

■両対数グラフ：

チェックを入れると、表示されるグラフが両対数表示になります。

■表示タイミング：

出力画面の更新タイミングを設定できます。

一定ステップ毎に更新させたい時は「指定シミュレーション間隔で表示」にチェックを入れてください。初期状態では１が入力されており、１ステップ実行される度に１回マップ出力画面が更新されます。

「シミュレーション終了時」にチェックを入れるとシミュレーション終了時に表示されます。

■背景色：

背景の色を変更することができます。

■軸ラベル：

グラフを表示する時にその軸のラベルとしてグラフとともに表示されます。

■グラフ要素リスト：

「追加」ボタンをクリックすると次の画面が表示されます。

■要素名：

このグラフ要素に任意の名称をつけます。

■プロット数：

表示させたいプロットの数を入力してください。指定された配列変数のうち、配列変数のインデックスのうち、０からここで指定した値までの値が描画されます。なお、ルール同様に変数や関数で指定することが可能です。（※参考：GetArraySize(【配列変数名】)で、引数に指定した配列変数のサイズを取得可能です）

■プロット値(X)を保持する配列変数名：プロット値(Y)を保持する配列変数名：

表示させるプロットの X の値と Y の値を決める変数を選択してください。配列の変数を指定することで、複数のプロットを行うことができます。

■線の太さ：

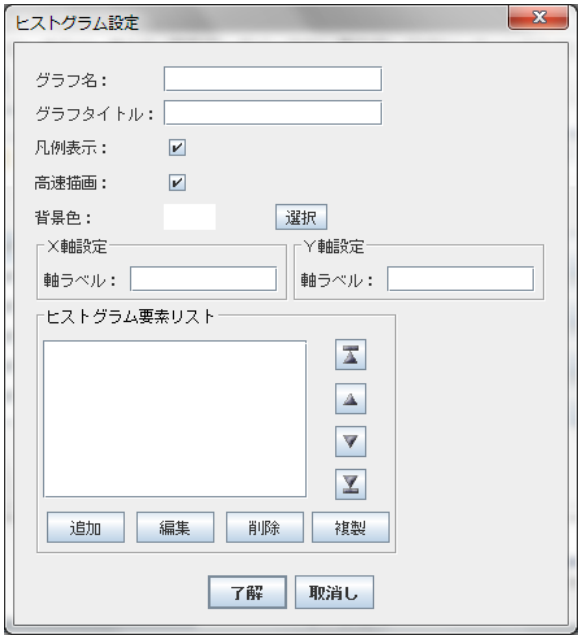
線の太さを変更できます。

■線の色：

線の色を変更できます。

2.4.3.8. ヒストグラム出力設定

コンポーネントをヒストグラムで表示する出力方法です。「追加する出力種類」で「ヒストグラム」を選択すると、次のダイアログが表示されます。なお、ヒストグラムはシミュレーション終了時に１度だけ描画されます。



■グラフ名：

このグラフを呼び出す時に参照する名称です。

■グラフタイトル：

出力画面にて表示されるグラフにこの「グラフタイトル」が表示されます。

■凡例表示：

ここにチェックを入れておくと、グラフの横に凡例が表示されます。

■高速描画：

グラフ表示を簡略化することで、artisoc の処理速度を高速化します。計算に時間がかかるときに使用してください。

■背景色：

背景の色を変更することができます。

■軸ラベル：

グラフを表示する時にその軸のラベルとしてグラフとともに表示されます。

■グラフ要素リスト：

「追加」ボタンをクリックすると次の画面が表示されます。

■要素名：

このグラフ要素に任意の名称をつけます。

■データ数：

ヒストグラムに表示させたいデータの数を入力してください。指定された配列変数のうち、配列変数のインデックスのうち、0からここで指定した値までの値が描画されます。なお、ルール同様に変数や関数で指定することが可能です。（※参考：GetArraySize(【配列変数名】)で、引数に指定した配列変数のサイズを取得可能です）

■区間数：

一つの棒として表す区間の総数を入力してください。

■値を保持する配列変数名：

ヒストグラムに反映するデータとなる配列変数を選択してください。

■棒の色：

棒の色を変更することができます。

※補足

ヒストグラムで表示する対象の変数は配列変数となります。例えば、Universe 以下に HistData(100):Integer という配列変数を定義し、HistData(0)～HistData(99)までの変数にそれぞれ何らかの値を入れます。これらのデータをもとに度数分布を表示します。設定方法は下記の通りです。

データ数：100

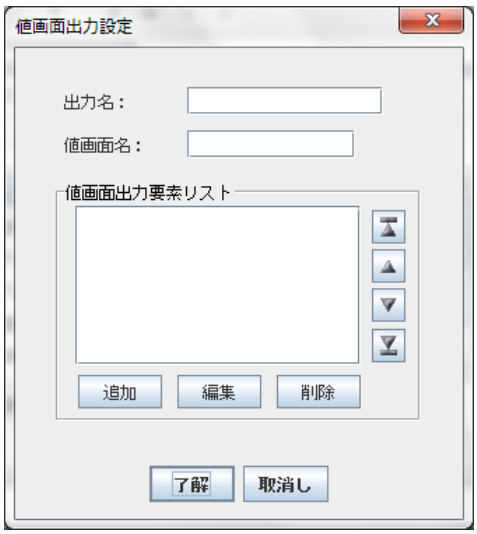
区間数：10 (HistData(0)～HistData(99)に格納してある値の最大値と最小値の区間を 10 分割します)

変数名：Universe.HistData

10 分割したそれぞれの範囲内でのデータ数を棒グラフで表現します。なお、ヒストグラムはシミュレーション終了時に 1 度だけ描画されます。

2.4.3.9. 値画面出力設定

変数の数値を表示する画面の出力方法です。「追加する出力種類」で「値画面出力」を選択すると、次のダイアログが表示されます。



■出力名：

ウィンドウの枠に表示されるタイトルです。

■値画面名：

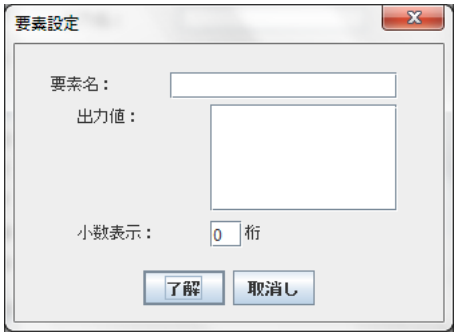
ウィンドウの中に表示されるタイトルです。

■値画面出力要素リスト：

値を表示する変数が表示されます。表示の順番は右側のボタンで操作することができます。

■値画面出力要素リストの追加

値画面出力要素リストの「追加」ボタンをクリックすると、次の画面が表示されます。



■要素名：

この数値画面出力要素に任意の名称をつけます。

■出力値：

出力したい変数や計算式を入力します。

■小数表示：

小数点以下何桁で表示するかを設定します。

もし、画面上に複数の要素を同時に表示したい場合は、それら要素毎に（ex.エージェント 1、エージェント 2、変数 1、etc.）数値画面出力要素として追加する必要があります。（つまり要素の数だけ追加リストに並ぶことになります）

「編集」ボタンをクリックすると、任意の既存の数値画面出力要素を編集することが出来ます。また「削除」ボタンによって、任意の既存の数値画面出力要素を削除することが出来ます。

2.4.3.10. ファイル出力設定

シミュレーションのログファイルをファイルに出力します。「追加する出力種類」で「ファイル出力」を選択すると、次のダイアログが表示されます。

■出力名：

このファイル出力を呼び出す時に参照する名称です。

■ファイル名：

出力するファイル名を指定します。

■出力間隔：

ファイルに出力するタイミングを設定します。

■区切り文字：

出力データの区切り文字を選択します。

■ファイル出力要素リストの追加

数値画面出力要素リストの「追加」ボタンをクリックすると、次の画面が表示されます。

■要素名：

このファイル出力要素に任意の名称をつけます。

■出力値：

出力したい変数を入力します。

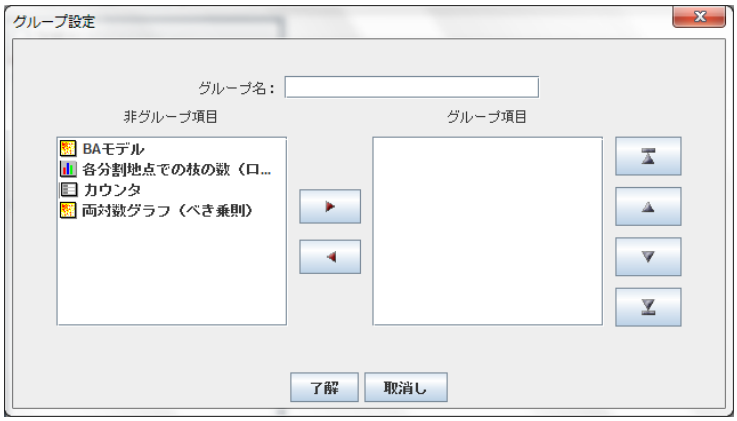
■小数表示：

小数点以下何桁で表示するかを設定します。

もしも、出力ファイルに複数のデータを同時に出力したい場合は、それら要素毎に（ex.エージェント 1、エージェント 2、変数 1、etc.）ファイル出力要素として追加する必要があります。（つまり要素の数だけ追加リストに並ぶことになります）
「編集」ボタンをクリックすると、任意の既存のファイル出力要素を編集することが出来ます。また「削除」ボタンによって、任意の既存のファイル出力要素を削除することが出来ます。

2.4.3.11. 出力項目グループ化設定

複数の出力設定項目をグループ化し１つの出力画面として出力します。「追加する出力種類」で「グループ」を選択すると、次のダイアログが表示されます。



■グループ名：

このグループ出力を呼び出す時に参照する名称です。グループ化した画面の画面名になります。

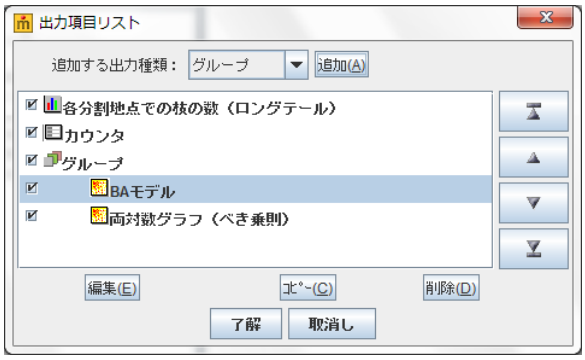
■非グループ項目：

現在グループ化されていない出力項目のリストが表示されます。グループ化対象の出力項目を選択し、リスト横の右矢印ボタンを選択することで、グループ項目に移行します。

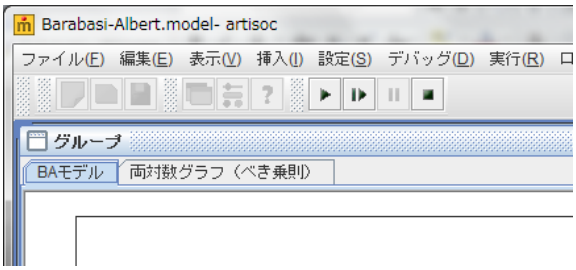
■グループ項目：

現在グループ化されている出力項目のリストが表示されます。グループ化解除対象の出力項目を選択し、リスト横の左矢印ボタンを選択することで、非グループ項目に移行します。また、グループ化表示画面の表示順番を上下のボタンを押下することで入れ替えることができます。

グループ化を行うと、下図のように出力設定画面上ではグループ化した項目がグループ名の下にインデントで表示されます。グループ内であっても、個別に出力項目の表示、非表示を出力設定画面上のチェックで指定することができます。なお、グループを削除した場合にも、グループ化した出力項目は削除されません。



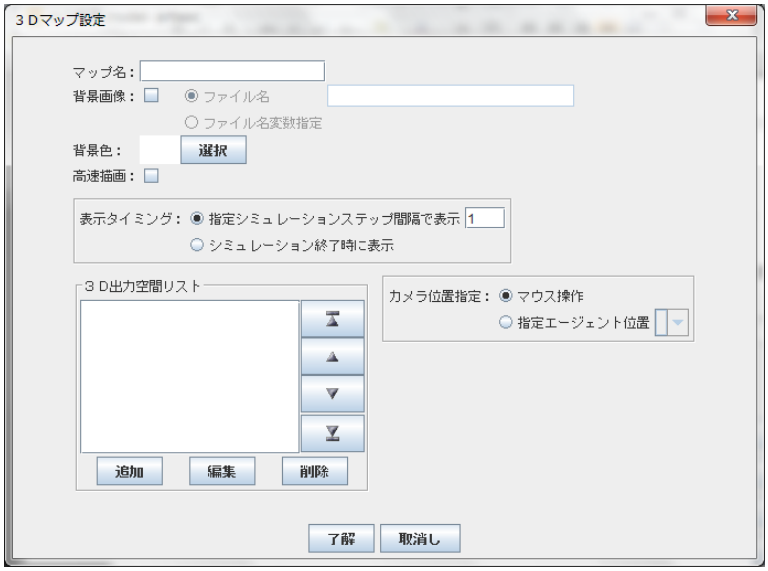
グループ化した画面は、シミュレーション実行時に下図のようにグループ名の１つの画面にグループ化した各出力項目がタブで表示されます。マウスでタブを切り替えることで、表示する出力項目を選択します。



2.4.3.12. ３Ｄマップ出力設定

空間を３次元表示するマップを出力します。3D マップは、指定された空間を OpenGL で 3D 描画します。お使いのＰＣのグラフィックス表示環境が artisoc の 3D 表示ライブラリに対応していない場合には、３Ｄマップが表示されない場合がございますのでご注意ください。

「追加する出力種類」で「３Ｄマップ」を選択すると、次のダイアログが表示されます。



■マップ名：

この３Ｄマップを呼び出す時に参照する名称です。表示される３Ｄマップ画面の画面名になります。

■背景画像：

３次元空間の周囲にマッピングする画像を指定します。画像を指定する場合には、チェックを入れてファイル名を指定します。なお、指定された画像は３次元空間上に“天球マッピング”の方法でマッピングされます。指定可能な画像形式は JPG,PNG です。

■背景色：

３次元空間の周囲の色を指定します。

■高速描画：

チェックを入れると、３次元表示要素（エージェント等）がワイヤフレームで表示され、面部分が描画されなくなります。これにより、高速に描画されます。

■表示タイミング：

出力画面の更新タイミングを設定できます。

一定ステップ毎に更新させたい時は「指定シミュレーション間隔で表示」にチェックを入れてください。初期状態では 1 が入力されており、1 ステップ実行される度に 1 回マップ出力画面が更新されます。

「シミュレーション終了時」にチェックを入れるとシミュレーション終了時に表示されます。

■カメラ位置指定：

3D マップ上に表示される視点を指定します。視点をマウスで操作指定する「マウス操作」と、指定したエージェントの視点で表示する「指定エージェント位置」のいずれかで指定します。

▽「マウス操作」の場合

右クリックしながら前後：Y 軸方向へ移動

右クリックしながら左右：X 軸方向へ移動

左クリックしながら前後：Z 軸方向へ移動

左クリックしながら左右：Direction の指定

中ボタンクリックしながら前後：アングルの指定

※マウスによっては中ボタンが存在しない場合があります。

※「マウス操作」の場合にはキーボードでも指定できます。

・キーボードでの視点操作

↑キー：アングル＋

↓キー：アングルー

←キー：Direction－

→キー：Direction＋

S キー：X 軸－

D キー：X 軸＋

E キー：Z 軸＋

X キー：Z 軸－

W キー：Y 軸＋

Q キー：Y 軸－

SPACE キー：Y 軸＋[高速]

RETURN キー：初期位置

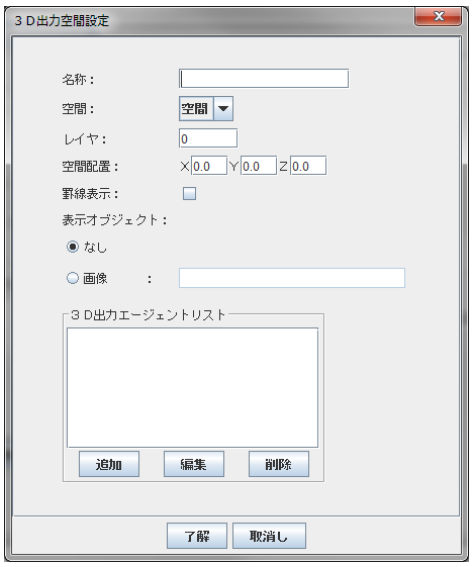
▽「指定エージェント位置」の場合

Universe 以下にある任意の「エージェント型」変数を指定可能です。このエージェント変数に代入されたエージェントの視点が表示されます。そのため、事前に Universe 以下に「エージェント型」変数を作成しておく必要があります。なお、マウスやキーボードによる視点の移動はできません。

■３D 出力空間リスト：

３D マップ上に表示する空間のリストです。同一の３D マップ上に、複数の空間（レイヤも含む）を表示することが可能となっております。ここで登録する空間は、モデルツリー上の「空間」のプロパティで「３D 表示対応」にチェックが入れてある必要があります。「追加」ボタンを押下することで、下記３D 出力空間設定ダイアログが表示されます。

▼３D 出力空間設定ダイアログ



■名称：

この３D 出力空間を呼び出す時に参照する名称です。

■空間：

対象とする空間をプルダウンから指定します。プルダウンからは、３D 表示対応の空間のみ表示されます。

■レイヤ：

３D マップ画面上に表示する上記指定した対象とする空間のレイヤの番号を指定します。

■空間配置：

指定空間（レイヤ）を３D マップ上に配置する座標を指定します。複数の空間（レイヤ）を３D マップ上に設定する場合には、各空間の位置関係をこの座標によって指定します。なお、各空間（レイヤ）上に存在するエージェントが表示される座標は、ここで指定された座標からの相対座標で表示されます。（例として、空間の配置座標を X：１０、Y：５、Z：0 と設定した場合、この空間に存在するエージェントの座標が X：２、Y：０、Z：１０のときは、３D マップ上にこのエージェントが描画される座標は、X：１２、Y：５、Z：１０となります。）

なお、3D マップ表示では左下が原点となります。

■罫線表示：

セル空間上に格子の罫線が表示されます。

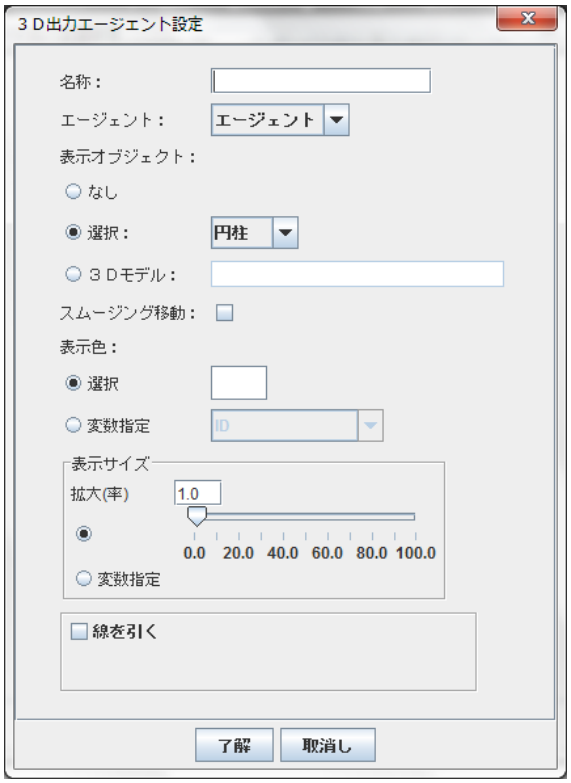
■表示オブジェクト：

表示対象の空間にマッピングする画像を指定します。指定可能な画像形式は、PNG と JPG です。

■３D 出力エージェントリスト：

この空間上に表示するエージェントのリストを表示します。「追加」ボタンを押下することで、下記ダイアログが表示され３D 出力対象のエージェントを追加します。

▼３D 出力エージェント設定ダイアログ



■名称：

この３D 出力エージェントを呼び出す時に参照する名称です。

■エージェント：

対象とするエージェントをプルダウンから選択します。

■表示オブジェクト：

「なし」、基本形状から選択する「選択」、ユーザの用意した３Dモデルファイルを指定する「３Dモデル」のいずれか指定可能です。

「選択」では、円柱、三角柱、四角柱、球、人型、車型から選択可能です。

「３Dモデル」では、3dMax 形式の.3ds ファイル形式のファイルが指定可能です(それ以外のフォーマットは対象外)。表示されるエージェントの位置は、３Dモデルの原点をセルの中心になるようになっております。３Dモデル作成時に、原点を中心としたモデルとなるようにしてください。

■スムージング移動：

チェックを入れると、ステップごとの座標間の移動が補完されスムーズに移動します。ただし、十分にシミュレーション速度が速い場合には、補完が追い付かず実際とは異なる位置に描画されてしまう可能性がありますのでご注意ください。シミュレーション実行速度の設定については、2.4.4 の実行環境設定ダイアログを参照してください。

■表示色：

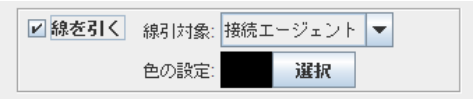
エージェントの形状が「選択」の基本形状をの場合に、基本形状の色を指定します。色の指定の仕方は、固定色または変数による指定のいずれかが選択可能です。変数による指定の場合には「変数指定」を選択後、色指定値を格納する変数をプルダウンより選択してください。

■表示サイズ：

エージェント表示オブジェクトのサイズを指定します。拡大率で指定します。固定値または変数による指定のいずれかが選択可能です。固定値の場合は、テキストボックスに拡大率を直接指定するか、スライダーバーで値を指定してください。変数による指定の場合には「変数指定」を選択後、拡大率指定値を格納する変数をプルダウンより選択してください。

■線を引く：

出力対象のエージェントがエージェント集合変数を持つ場合は、出力対象エージェントとエージェント集合の要素それぞれとを線で結ぶことができます。線を引くにチェックを入れると、線引き対象、色の設定を選択できるようになります。

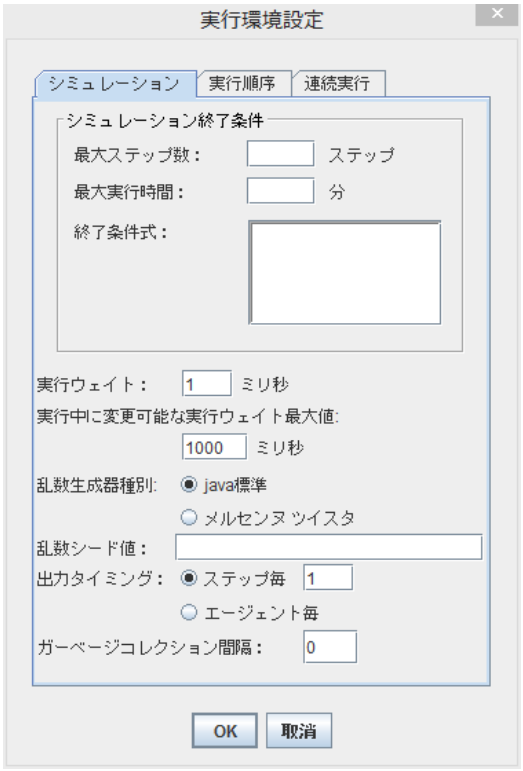


2.4.4. 実行環境設定ダイアログ

ここでは、シミュレーションを実行する上での各種設定を行ないます。

「設定」メニューから「実行環境設定」を選ぶと、「実行環境設定ダイアログ」が表示されます。このダイアログには「シミュレーション」「実行順序」「連続実行」の３つのタブがあり、切り替えて設定することが出来ます。

シミュレーションタブ



■最大ステップ数：

途中でユーザがシミュレーションを停止しない限り、この数値の回数だけステップを実行します。

■最大実行時間：

途中でユーザがシミュレーションを停止しない限り、この数値の時間が経過するまでステップを実行します。

■終了条件式：

途中でユーザがシミュレーションを停止しない限り、ここに入力された条件式が成立するまでステップを実行します。

■実行ウェイト：

1 ステップ終了した段階で、この数値の時間だけ、処理をウェイトします。処理の速いマシンなどでシミュレーション速度が速

すぎる場合などに、適当な数値を入力してください。

■実行中に変更可能な実行ウェイト最大値：

ツールバー上でスライダーバーにて変更可能な実行ウェイトの最大値を指定します。

■乱数生成器種別：

乱数生成アルゴリズムを指定することが出来ます。Java 標準の乱数発生器（擬似乱数発生器：PRNG)）とメルセンヌツイスターを選択可能です。デフォルトは Java 標準の乱数発生器が使用されます。

■乱数シード値：

乱数のパターンを指定することが出来ます。

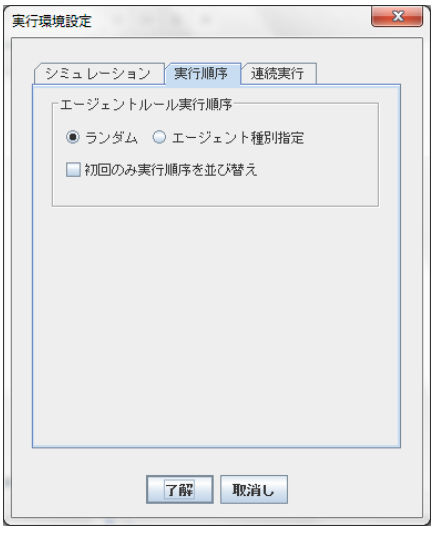
■出力タイミング：

画面への出力タイミングを設定します。「ステップ毎」を選択した場合、指定したステップ毎に画面が描きかえられます。「エージェント毎」を選択した場合、1 つのエージェントがルールを実行する毎に画面が描きかえられます。

■ガーベージコレクション間隔：

画面出力をスムーズに行うため、Java のガーベージコレクションのタイミングを明示的に指定することができます。デフォルトでは 10 が指定されており、10 ステップに 1 回ガーベージコレクションが行われます。
0 を指定したときは明示的にガーベージコレクションを行なわないため、最高速度で実行できます。

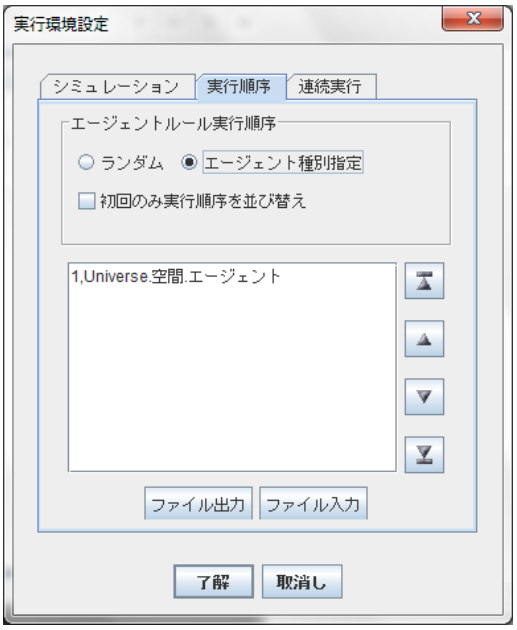
実行順序タブ（ランダム）



■エージェントルール実行順序：

Universe の Univ_Step_Begin はそのステップの最初に実行され、Univ_Step_End はそのステップの最後に実行されますが、エージェントに関しては、この画面にて設定を変えることが出来ます。エージェントルール実行順序が「ランダム」のときは、各ステップでランダムにエージェントの実行順序が入れ替わります。また、「初回のみ実行順序を並び替え」をチェックするとシミュレーション開始時のみ 1 回だけ実行順序が並べ替えされます。

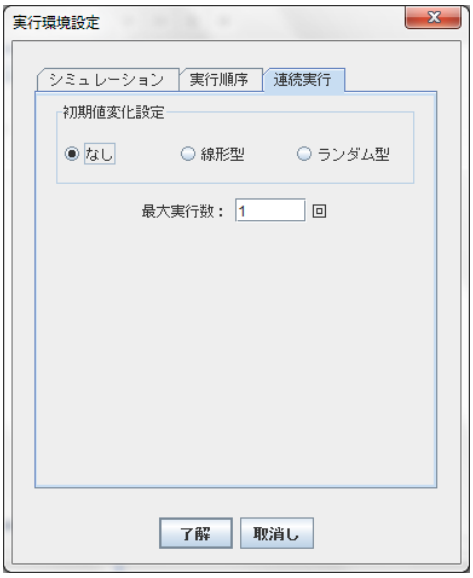
実行順序タブ（エージェント指定）



■エージェントルール実行順序：

エージェントルール実行順序が「エージェント種別指定」のときは、各ステップでエージェント種別毎に優先順位を持った実行順序となります。すべてのエージェントの優先順位を「1」としたとき、「ランダム」を指定したときと同じになります。 1 種類のエージェントのみ優先順位を「2」にしたときは、優先順位が「1」のエージェントが実行された後に優先順位「2」のエージェントが実行されます。なお、各ステップで優先順位毎にランダムにエージェントの実行順序が入れ替わります。また、「初回のみ実行順序を並び替え」をチェックするとシミュレーション開始時のみ 1 回だけ実行順序が並べ替えされます。

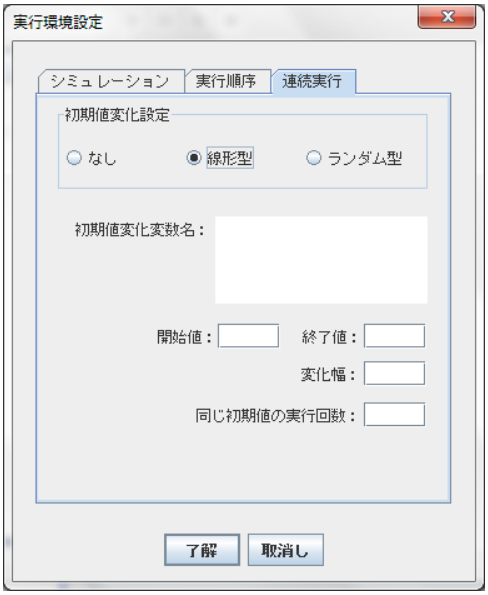
連続実行タブ　（初期値変化設定：なし）



■初期値変化設定：

初期値変化設定が「なし」のときは、最大実行数を回数で指定できます。

連続実行タブ　（初期値変化設定：線形型）



初期値変化設定が「線形型」のときは、次のパラメータを指定できます。

■初期値変化変数名：

初期値を変化させたい変数名を指定します。

■開始値：

変化する初期値の開始値を指定します。

■終了値：

変化する初期値の終了値を指定します。

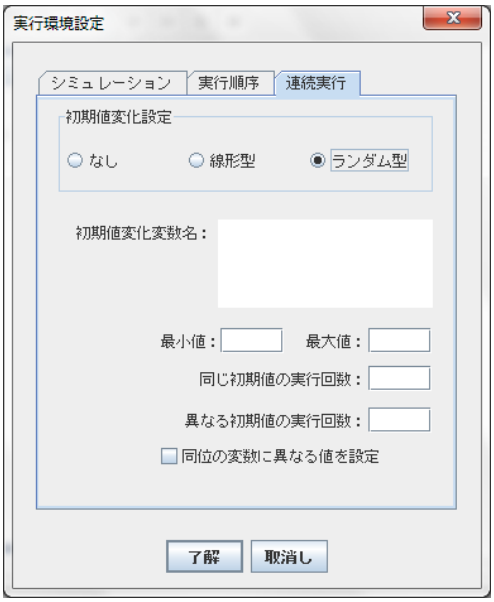
■変化幅：

変化する初期値の変化幅を指定します。

■同じ初期値の実行回数：

同じ初期値で実行するときの回数を指定します。

連続実行タブ　（初期値変化設定：ランダム型）



初期値変化設定が「ランダム型」のときは、次のパラメータを指定できます。

■初期値変化変数名：

初期値を変化させたい変数名を指定します。

■最小値：

変化させる変数の初期値の最小値を指定します。

■最大値：

変化させる変数の初期値の最大値を指定します。

■同じ初期値の実行回数：

初期値を変化させる前に、同じシミュレーションを何回実行させるかを設定します。

■異なる初期値の実行回数：

初期値を乱数で変化させたシミュレーションを何回実行させるかを設定します。

■同位の変数に異なる値を設定：

同じエージェント（例：カメ(0)、カメ(1)、カメ(2)・・・）が持つ同じ変数に、同じ乱数の値を当てはめる場合は、ここにチェックをいれます。

2.4.5. コントロールパネル設定

Universe の直下にある変数については実行中にリアルタイムに変更することが出来ます。また、マウスやキーボードを用いて、ある特定のルールを実行することも可能です。

コントロールパネル設定において、変化させる変数を指定している場合、以下のようなコントロールパネルが追加されます。

■ボタンの時：

ボタンを押下時に「on」の状態になり、対応する変数を任意の値にします。

■トグルボタンの時：

ボタンを押下時「on」の状態を保持し、もう一度押すと「off」の状態を保持します。「on」と「off」それぞれの時に、対応する変数の値を設定した値にします。

■スライドバーの時：

コントロールパネル設定ダイアログに入力した「範囲」内をマウスのドラッグによって任意に動かすことが出来ます。

■直接入力の時：

マウスでクリックして選択した後、キーボードで直接数値等を入力します。

■ドロップダウンリストの時：

表示セルの右ボタンをクリックすることでリストの一覧が表示され、その中から選択した値を変数に代入します。

コントロールパネルの設定方法

コントロールパネルを設定するには「設定」メニューから「コントロールパネル設定」を選択すると、次のダイアログが表示されます。

■追加：

コントロールパネルに表示させる設定項目を追加します。

■編集：

既にある設定項目を編集します。

■削除：

既にある設定項目を削除します。

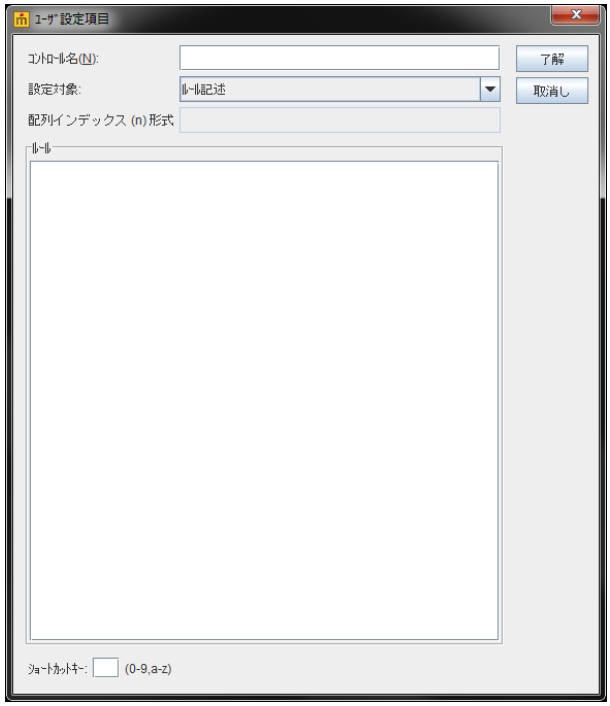
「追加」や「編集」をクリックすると次のような「ユーザ設定項目」ダイアログが開きます。

（1）設定対象：ルール記述

設定対象として「ルール記述」を選択した場合、コントロールパネルにボタンが追加されます。このボタンを押した場合、記述したルールが実行されます。ショートカットキーを設定すれば、コントロールパネルのボタンを押すことなく、キーボードの入力で特定のルールを動作させることが可能です。

■パネル種別：

「ダイアログ」を選択した場合にはコントロールパネルをダイアログとして別画面で表示されます。「固定」を選択した場合にはartisocのメイン画面の左側にコントロールパネルが埋め込まれ固定されます。



■コントロール名:

押すとルール記述を実行するボタンに表示される名前です。分かりやすい任意の名前を付けてください。

（2）設定対象：ボタン、トグルボタン、スライドバー、直接入力

■コントロール名:

押すとルール記述を実行するボタンに表示される名前です。分かりやすい任意の名前を付けてください。

■設定対象：

コントロールパネルで操作したい変数を選択します。ただし、Universe の直下にある変数しか選択できません。

■インターフェース＞ボタン：

On の時（クリックした時）だけ任意の値を代入したい時選択します。

●ＯＮの値：

ＯＮにしたときに変数に代入する値を入力してください。

■インターフェース＞トグルボタン：

On の時と Off の時、それぞれに任意の値を代入したい時選択します。

●ＯＮの値：

ＯＮにしたときに変数に代入する値を入力してください。

●ＯＦＦの値：

ＯＦＦにしたときに変数を代入する値を入力してください。

■インターフェース＞スライドバー：

スライドバーを左右にドラッグすることにより、任意の値の範囲で断続的に変化させたい時選択します。

●範囲：

左に変数に代入したい値の最小値を、右に最大値を入力してください。

●目盛り間隔：

ここに入力した値が断続的な変化の最小間隔になります。

■インターフェース＞直接入力：

変数へ代入する値を直接入力したい時に選択します。

■インターフェース＞ドロップダウンリスト：

変数へ代入する値をドロップダウンリストから選びたい時に選択します。

●リスト項目：

選択候補をカンマ区切りで入力してください。

例） 晴れ, 雨, 雪, 曇

2.5. シミュレーション実行機能

ユーザがシミュレーションを開始したり、途中で一時停止したり、終了させたりする場合に用います。

シミュレーションを実行させるには、次の実行パネルを使います。

網掛け部分をドラッグ＆ドロップすることにより、実行パネルを独立に表示させることもできます。

「実行」ボタン：

シミュレーションを開始する時に、クリックします。

「ステップ実行」ボタン：

シミュレーションを 1 ステップ分だけ実行します。続けてクリックすれば、その都度 1 ステップずつ実行して、その後一時停止状態になります。

「一時停止」ボタン：

シミュレーションを一時停止させる時に、クリックします。

「停止」ボタン：

シミュレーションを終了させる時に、クリックします。

なお、[ESC]キーをクリックしても同様にシミュレーションが停止します。

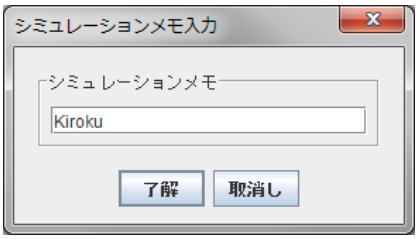
2.6. ログ再生機能

プレゼンテーション等でシミュレーションの様子を表示する際は、その場でシミュレーションを実行するよりも、既に実行したシミュレーションを再生するほうが、スムーズに画面描画を行うことができます。また、任意のステップ状態を表示したり、逆再生を行うことも可能です。

この機能を使用するには、まず、あらかじめ設定されたシミュレーションを実行し、記録しておく必要があります。

2.6.1. ログの記録

「実行」メニューの「記録して実行」を選択すると、「シミュレーションメモ入力」ダイアログが表示されます。

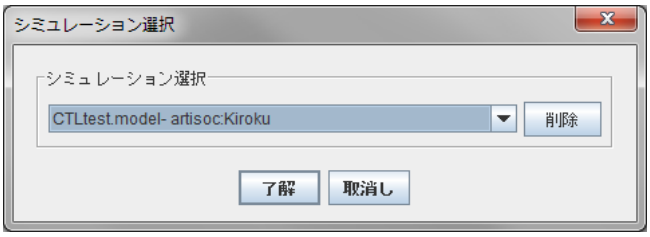


再生するシミュレーションを識別するためのメモを入力し「了解」ボタンを押すと、シミュレーションの実行と同時に記録が開始されます。「実行」メニューの「終了」を選択すると、シミュレーションを終了し、シミュレーションの記録が完了します。

2.6.2. ログの再生と削除

ログ再生モードに移行すると現在のモデルが失われることがあります。ログ再生モードに移行する前に、必ず現在のモデルを保存してください。

「ログ」メニューの「再生」を選択すると、「シミュレーション選択」ダイアログが出現します。

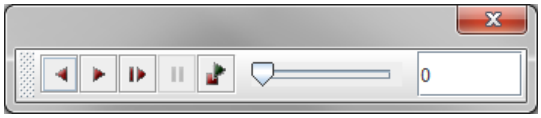


記録したシミュレーションは「(モデルファイル名) - artisoc: (シミュレーションメモ))」という名前で記録されています。プルダウンリストから再生したいシミュレーションを選択し「了解」ボタンを押すと、シミュレーションの再生が可能になります。削除したいシミュレーションのログがある場合は、そのシミュレーションを選択し「削除」ボタンを押すと、その記録が削除されます。

再生パネル

再生したいシミュレーションの選択までが済むとメニューバーの下に「再生パネル」が現れます。「実行パネル」によく似たボタン類と、スライドバー、表示ステップで構成されています。実行パネル同様、網掛け部分をドラッグ＆ドロップすることによ

り、再生パネルも独立させることが可能です。



再生パネル（独立状態）

「逆再生」ボタン：

最後から巻き戻しながら連続して再生するボタンです。「ログ」メニューの「逆再生」と同じです。

「再生」ボタン：

最後まで連続して再生するボタンです。「ログ」メニューの「再生」と同じ機能です。

「ステップ再生」ボタン：

押すたびに 1 ステップずつ再生するボタンです。

「一時停止」ボタン：

再生中に押すと一時停止します。「ログ」メニューの「一時停止」と同じ機能です。

「停止」ボタン：

ログ再生モードを停止するボタンです。ログ再生モードを閉じ、新規作成を行った直後の状態になります。「ログ」メニューの「停止」と同じ機能です。

■スライドバー： 

スライドバーをドラッグすることにより、表示ステップを自由に動かすことができます。

■表示ステップ：

画面に出力しているステップを表示します。ここに値を直接入力し、エンターキーを押すことにより、指定したステップの状態を画面に出力することもできます。

2.7. ウィンドウ整列表示機能

■重ねて表示：

「ウィンドウ」メニューから「重ねて表示」を選択すると、表示しているウィンドウをカスケード（重ねて）表示ができます。

■並べて表示：

「ウィンドウ」メニューから「並べて表示」を選択すると、表示しているウィンドウをタイル状に並べて表示ができます。

2.8. コンソール画面機能

コンソール画面の出力

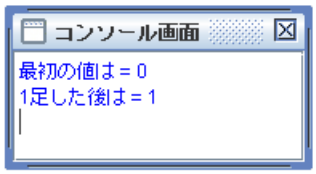
ルールの中に、Print 関数または PrintLn 関数を書くことにより、コンソール画面に値を出力することが出来ます。Print 関数と PrintLn 関数の違いは、改行を行うかどうかの違いです。

書式： PrintLn(引数)　※引数が文字列の場合は”” で囲む

例：

```
Agt_Init{
  Dim X as Integer
  X = 0
  PrintLn ( "最初の値は = " & X )
  X = X + 1
  PrintLn ( "1 足した後は = " & X )
}
```

実行結果

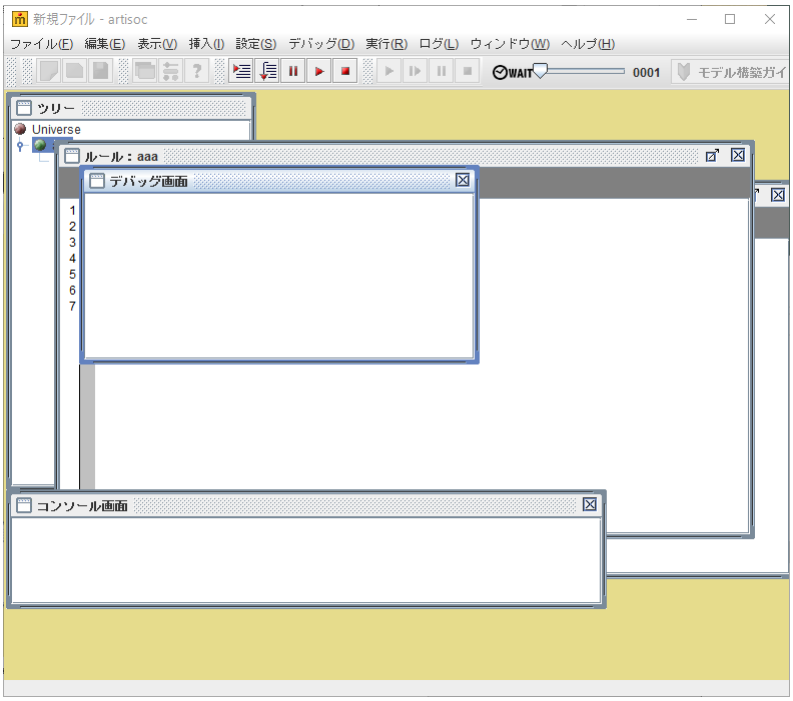


2.9. デバッグ機能

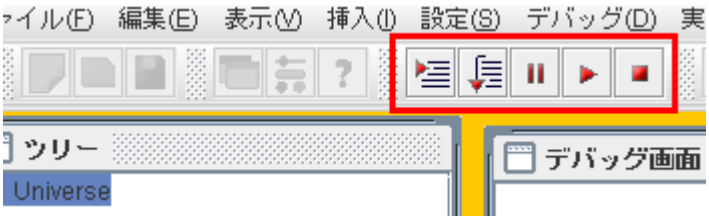
エージェントルールの間違いをチェックしたり、代入された変数を表示したりするときはデバッグ機能を利用します。デバッグモードでは、ルールを一行ごとに実行することができます。

2.9.1. デバッグモードでの操作

「デバッグ」メニューの「デバッガの起動」を選択すると、画面がデバッグモードに変わります。



デバッグを行うためには、次のデバッグパネルを使います。



網掛け部分をドラッグ＆ドロップすることにより、デバッグパネルを独立させることもできます。


「ステップイン」ボタン：

ステップインする時に、クリックします。関数の中に入って次の行をデバッグするときに利用します。

 **「ステップオーバー」ボタン：**

ステップオーバーする時に、クリックします。関数の中に入らずに次の行をデバッグするときに利用します。

 **「一時停止」ボタン：**

一時停止する時に、クリックします。

 **「続けて実行」ボタン：**

次のブレークポイントまで実行する時に、クリックします。

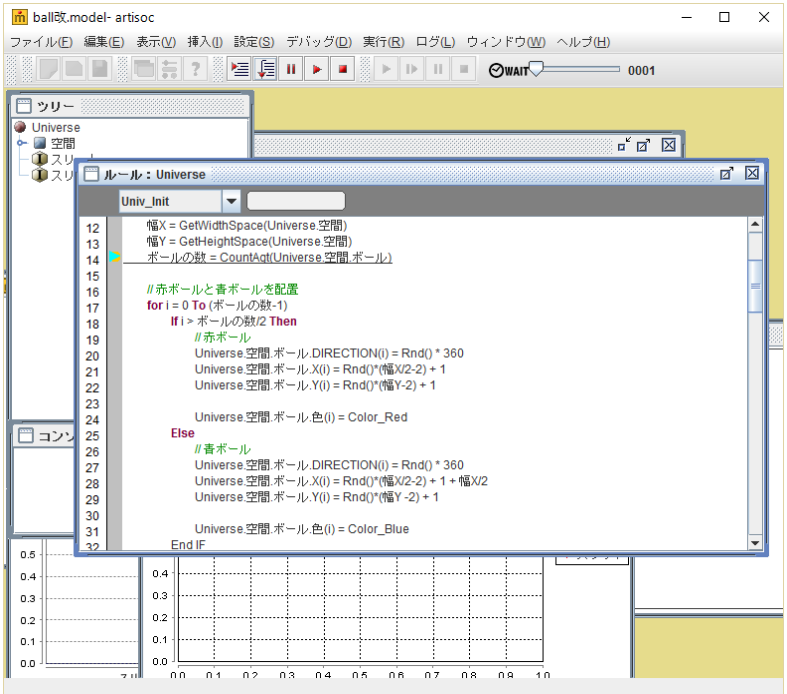
 **「デバッガの終了」ボタン：**

デバッグモードを終了する時に、クリックします。

2.9.2. ブレークポイントの設定と解除

ルールエディタの指定した行で停止したいときにはブレークポイントを設定します。

ルールエディタを表示して右クリックし、「ブレークポイントを設定／解除」を選択すると、ブレークポイントを設定することが出来ます。

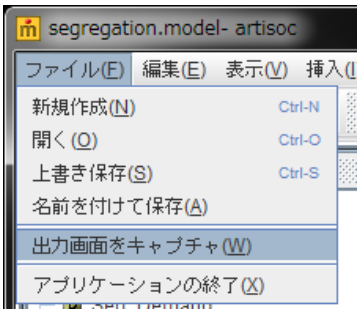


また、ブレークポイントが設定されている行で同様に「ブレークポイントを設定／解除」を選択すると、ブレークポイントの設定が解除されます。

2.10. 出力画面のキャプチャ機能

出力画面をキャプチャすることができます。

シミュレーションを一次停止している状態で任意の出力画面を選択し、ファイルメニューの「出力画面をキャプチャ」を選択すると、出力画面をキャプチャすることができます。



キャプチャした画像は、クリップボードにコピーされるため他のワープロソフトやプレゼンテーションソフトに貼り付けることができます。なお、この機能での３Ｄマップ画面のキャプチャはできません。OS のスクリーンショット機能をご利用ください。

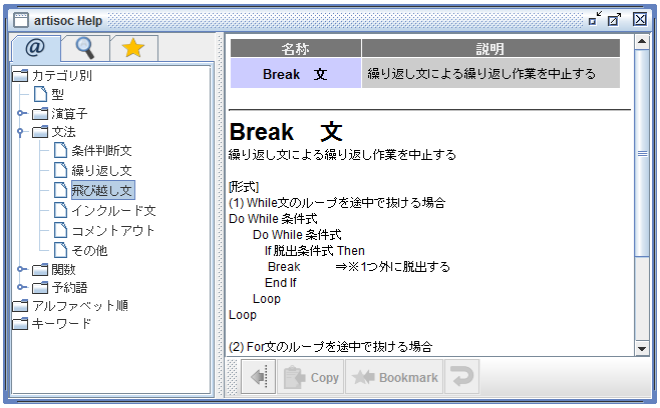
2. 11. ヘルプ機能

artisoc には、ヘルプが内蔵されており、関数、文法、演算子、予約語、型といった要素とその要素のヘルプを参照することができます。ヘルプを開くには、メニューバーの「ヘルプ」から artisoc ヘルプを選択するか、上部のパネルのヘルプボタンをクリックしてください。

2. 11. 1.　　カテゴリ別索引

カテゴリ別索引では、カテゴリ別に整理されたツリーを辿っていくことで、目的の要素とそのヘルプを探し出すことができます。探したい要素のカテゴリを選択していくことで、要素の候補が絞り込まれていきます。また、アルファベット順にソートされた一覧を利用することもできます。

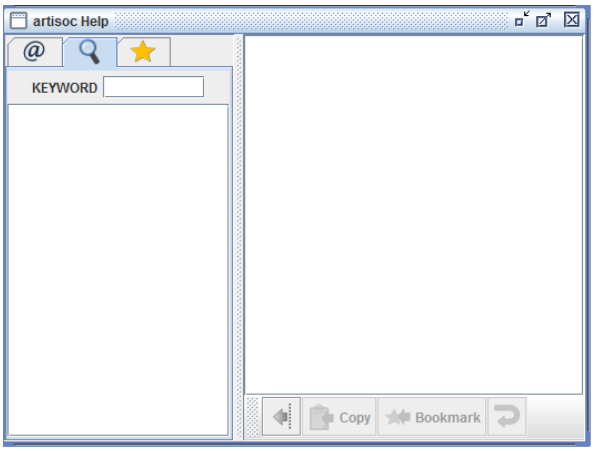
カテゴリ別索引は、artisoc ヘルプ内の、@マークのタブを開くと実行できます。



2. 11. 2.　　キーワード検索

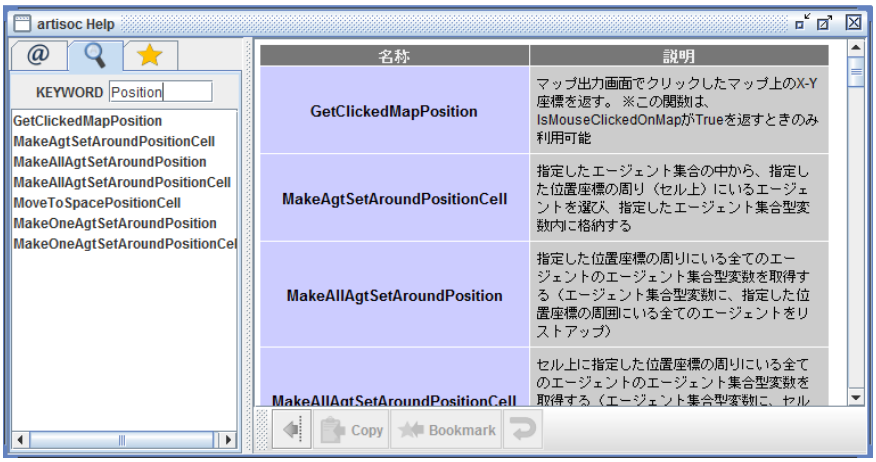
キーワード検索することにより、キーワードを含む要素とそのヘルプを呼び出すことができます。

キーワード検索は、artisoc ヘルプ内の、虫めがねのマークのタブを開くと実行できます。



KEYWORD に単語を入力し、Enter キーを押すとその単語の文字列が含まれる要素が

表示されます。



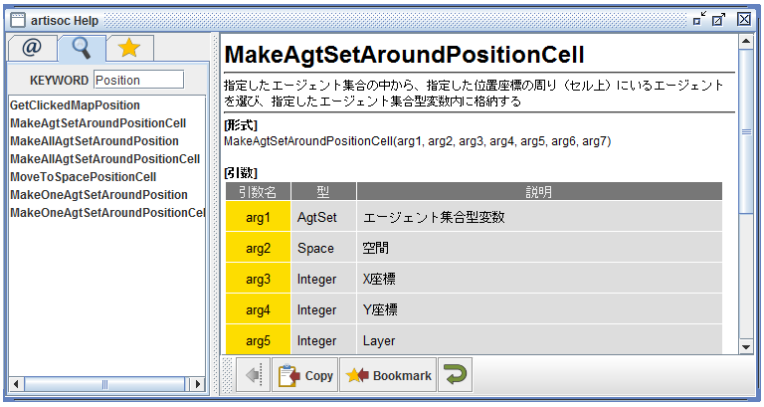
右のウィンドウのリンクをクリックすることで目的の要素のヘルプを参照することができます。

2. 11. 3.　　お気に入り登録

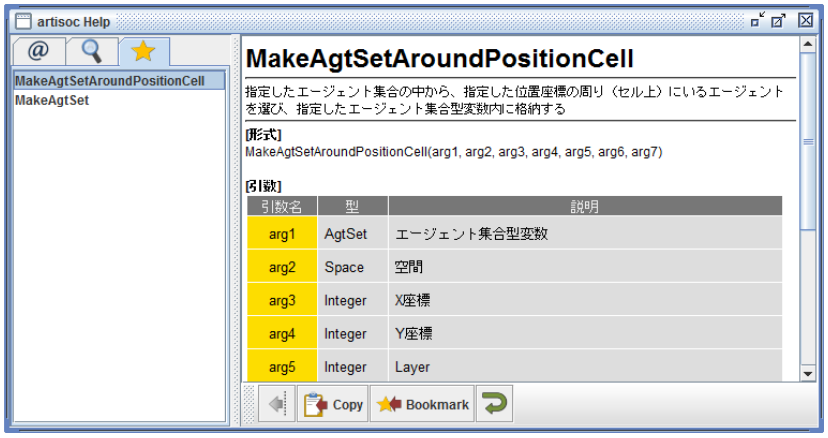
お気に入りを登録することにより、よく参照したいヘルプを簡単に呼び出すことができるようになります。

お気に入りは、artisoc ヘルプ内の、星のマークのタブを開くと表示されます。

要素のヘルプを参照した際に、「Bookmark」をクリックすることでお気に入りに登録できます。



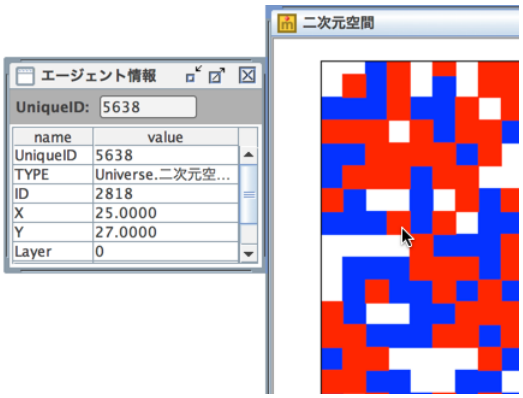
以下のようにお気に入りに登録されます。



2. 12. エージェント情報表示機能

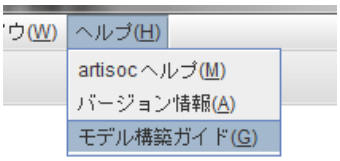
シミュレーション実行中における artisoc 上の指定されたエージェントの持つ属性情報を表示する機能です。ツールバーのエージェント情報表示切替ボタンを押下すると、エージェント情報画面が表示されます。

エージェント情報表示画面上に表示対象のエージェントのユニーク ID を入力するか、2 次元マップ出力画面に表示されているエージェントを選択し右クリックすることで、指定したエージェントの属性情報一覧が表示されます。シミュレーション実行時に該当エージェントの属性値の変化に合わせて、表示内容も自動で更新されます。

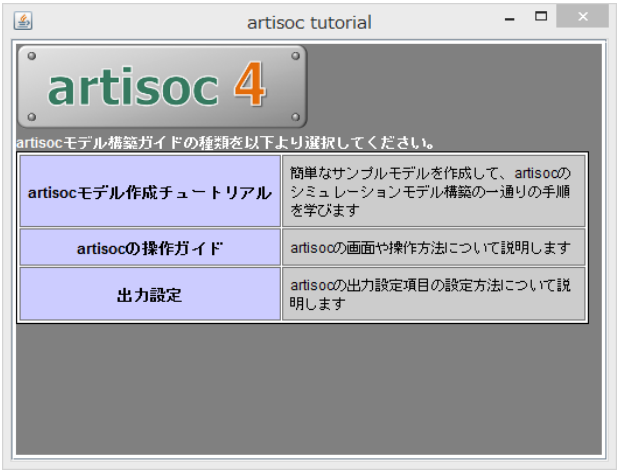


2. 13. モデル構築ガイド機能

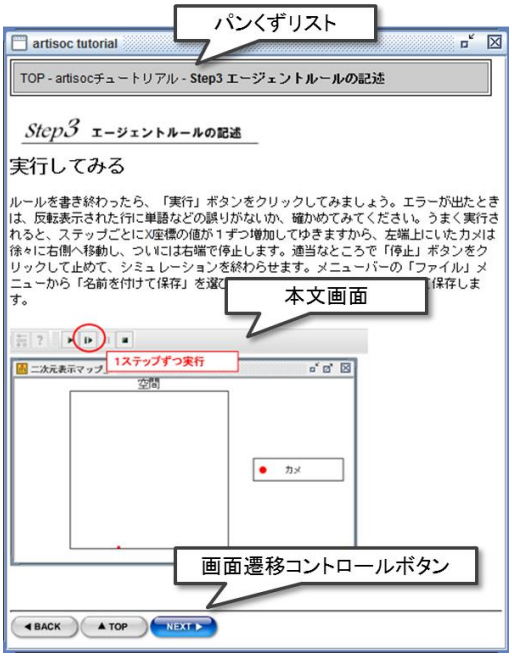
モデルを構築するにあたって、有用な情報をツール上で扱えるようにモデル構築ガイド機能を搭載しております。artisoc の「ヘルプ」メニューから「モデル構築ガイド」を選択してください。



artisoc モデル構築ガイドの画面が表示されます。ガイドの種類を目的に応じて選択してください。



各ガイドの画面構成は下図のようになっています。



2. 14. バージョン情報表示と更新確認機能

「ヘルプ」メニューから「バージョン情報」を選択すると、バージョン情報、コピーライトおよびライセンス期限が表示されます。

更新を確認

「更新を確認」のボタンを押し更新確認機能を使用することにより、最新の artisoc がリリースされているか確認することができます。（※更新確認機能を利用する際には、お使いのＰＣをインターネットに接続している必要があります。なお、インターネットへのプロキシ接続など環境によってはこの機能をご利用いただけない場合がございます。プロキシ経由で、インターネットへ接続して更新を確認する場合には「2. 15 **プロキシの設定**」をご参照ください。）

最新の artisoc がリリースされている場合は、最新版のリリースされている旨のダイアログが表示されます。「はい」ボタンを押下することで、ダウンロードサイトが表示されますので、ユーザ登録時に発行されたアカウントでログイン後、最新版をダウンロードしてください。

ライセンス更新

「ライセンス更新」ボタンを押下すると、更新するライセンスの KeyCode を入力することで artisoc のライセンスを更新することができます。（※ライセンス更新機能を利用する際には、お使いのＰＣをインターネットに接続している必要があります。なお、インターネットへのプロキシ接続など環境によってはこの機能をご利用いただけない場合がございます。プロキシ経由で、インターネットへ接続して更新を確認する場合には「2. 15 **プロキシの設定**」をご参照ください。）

2.15. プロキシの設定

ネットワーク認証、更新情報確認、ライセンスの更新にはインターネットに接続された状態で行う必要があります。環境によっては、プロキシを介してインターネット接続する場合があります。その場合、認証に失敗する場合があります。その場合には、プロキシサーバを設定した上で操作する必要があります。設定は以下の手順で行います。

① テキストファイルを作成し、ファイルをテキストエディタで開きます。

② プロキシの設定をファイルに記述します。プロキシ設定を下記のとおり１行記入します。

```
http://[プロキシサーバ名]:[プロキシポート番号]
```

③ ファイル名を「proxy.conf」として保存し、ファイルを以下にコピーします。(管理者権限を要求される場合があります)

- Windows の場合: C:¥Program Files¥KOZO KEIKAKU ENGINEERING Inc¥artisoc 4.0 standard (64bit)
- MacOSX の場合： artisoc.app が配置されているフォルダ

④ artisoc を再起動します。

3. エージェントルール文法

第3章　エージェントルール文法

この項目は、マルチエージェントシミュレータのエージェントルール内で記述される言語についての仕様書です。プログラミング経験の無い人にとっては、少々敷居が高いかもしれませんが、市販されている文法書と合わせて研究してみてください。なお、ルールに使われている言語は基本的に Microsoft 社の Visual Basic に沿っています。

3.1. エージェントルールの全体構成

エージェントルールはエージェント毎に以下の構成で記述します。

ルールは関数の集まりです。

```
Agt_Init{
  ローカル宣言部
  実行部
}
Agt_Step{
  ローカル宣言部
  実行部
}
ユーザ定義関数※ {

}
```

＜注意＞：ユーザ定義関数は必ずしも必要ありません。

Universe では Agt_Init、Agt_Step ではなく Univ_Init、Univ_Step_Begin、Univ_Step_End、Univ_Finish を用います。

3.2. 特殊関数

■Agt_Init：

シミュレーションの実行に際して、エージェントが生成されるとき一回だけ実行される関数です。Agt_Init という名称は特別の意味を持っており、他の目的で使用することはできない予約語です。

以下の形式で記述します。

```
Agt_Init{
  変数の型宣言部
  実行部
}
```

■Agt_Step：

シミュレーションが終了するまで 1 ステップ毎に 1 回繰り返し実行される関数です。Agt_Step という名称は特別の意味をもっており、他の目的で使用することはできない予約語です。

以下の形式で記述します。

```
Agt_Step{
  変数の型宣言部
  実行部
}
```

■Univ_Init：

Universe のルール内で用いる関数で、シミュレーションの最初に 1 度だけ実行される関数です。Univ_Init という名称は特別の意味をもっており、他の目的で使用することはできない予約語です。

以下の形式で記述します。

```
Univ_Init{
  変数の型宣言部
  実行部
}
```

■Univ_Step_Begin：

Universe のルール内で用いる関数で、シミュレーションが終了するまで 1 ステップ 1 回繰り返し実行される関数です。各ステ

ップで、他のどのルールよりも最初に実行されます。Univ_Step_Begin という名称は特別の意味をもっており、他の目的で使用することはできない予約語です。

以下の形式で記述します。

```
Univ_Step_Begin{
  変数の型宣言部
  実行部
}
```

■Univ_Step_End：

Universe のルール内で用いる関数で、シミュレーション終了するまで繰り返し実行される関数です。各ステップで、他のどのルールよりも最後に実行されます。Univ_Step_End という名称は特別の意味をもっており、他の目的で使用することはできない予約語です。

以下の形式で記述します。

```
Univ_Step_End{
  変数の型宣言部
  実行部
}
```

■Univ_Finish：

Universe のルール内で用いる関数で、シミュレーションが終了する直前に 1 度だけ実行される関数です。Univ_Finish という名称は特別の意味をもっており、他の目的で使用することはできない予約語です。

以下の形式で記述します。

```
Univ_Finish{
  変数の型宣言部
  実行部
}
```

3.3. ユーザ定義関数

ユーザが自由に定義できる関数です。戻り値を返す『Function 型』と戻り値を返さない『Sub 型』ここで定義した関数は、変数の型宣言部で宣言した名前定数と同様に、当該エージェントルール内の Agt_Init 関数，Agt_Step 関数，ユーザ定義関数から呼び出すことができますが、他のエージェントのエージェントルールから呼び出すことはできません。

『Function 型』は以下の形式で記述します。

```
Function　関数名（パラメータ宣言）　As　関数戻り値の型
{
  変数の型宣言部
  実行部
  Return(式)
}
```

■Function：

ユーザ定義関数がここから定義される、ということを宣言します。

■関数名：

ユーザが自由に命名することができます。ただし、予約語や既に宣言した変数名や名前定数を使用することはできません。

■As　関数戻り値の型：

関数が値を返す必要がある場合に記述します。省略した場合、関数は値を返すことができません。関数戻り値の型には、関数が返す値の型を指定します。指定方法は変数の型の場合と同じです。

『Sub 型』は以下のように記述します

```
Sub　関数名（パラメータ宣言）
{
  変数の型宣言部
  実行部
}
```

■Sub：

ユーザ定義関数がここから定義される、ということを宣言します。

■関数名：

ユーザが自由に命名することができます。ただし、予約語や既に宣言した変数名や名前定数を使用することはできません。

■パラメータ宣言

関数に引き渡すパラメータがある場合には以下の形式で宣言します。

変数指定子１　As　変数の型,...，変数指定子ｎ　As　変数の型

(例)

```
Function function1(a as Integer, b As Integer, c As Double) As Integer
{
}
Sub sub1(a as Integer, b As Integer, c As Double)
{
}
```

＜注＞：Function 型と Sub 型の違いについて

ユーザ定義関数が実行された時に、その結果の値（Return 値）を直接実行文で使用したい時には Function を、そうでない場合は Sub を使います。例えば、

```
Agt_Step{
  Dim a As Integer
  a = Function1 (5,3)
}

Function function1(b As Integer, c As Integer) As Integer
{
  Dim answer1 As Integer
  answer1 = b + c
  Return (answer1)
}
```

というルールが記述してあったとすると、まず二行目では、Agt_Step のなかで「a」という整数型の変数が宣言されています。そしてその次に「function１」というユーザ定義関数に 5 と 3 という値を渡して計算させて、その結果を「a」に代入しなさい、と書かれています。

そしてユーザ定義関数の中では（5 行目から)、「answer1」という整数型の変数を宣言し（6 行目)、その変数に「b+c」（この場合は 5+3 になります) の計算結果を代入し（7 行目)、そして answer１の中身の値を Agt_Step の Function1 の所に返しなさい、と書かれています。

つまり、この例では 5+3 という計算の答えを「a」という変数に代入しなさい、ということが記述されているのです。しかし、もし、「a」という変数に計算結果を代入する必要が無ければ、ユーザ定義関数は値を返す必要は無くなります。その場合は「Function」ではなく「Sub」を使います。

3.4. 名前のきまり

エージェントルール内で使用する変数名、関数名は、以下の規則に従う必要があります。

- **名前の１文字目は英字、_（アンダースコア）、ひらがな、カタカナまたは漢字（空白を除く）である事。**
- **２文字目以降は、英字、数字、_（アンダースコア）、ひらがな、カタカナ、漢字（空白を除く）のいずれかである事。**
- **名前は 256 文字以内である事。**
- **名前に予約語を使用しない事。**

＜注意＞：英字の大文字／小文字は区別されません。

(例）以下の名前は同一とみなされます。

SUM, Sum, sum

（正しい名前の例）

Sum, sum_x, sum100

（誤った名前の例）

- **1x　　1 文字目に数字は使えません。**
- **x$　　2 文字目に特殊記号が使われています。**
- **next　　名前に予約語を使用しています。**

3.5. 変数宣言部

その関数の中だけで使用する変数を宣言する部分であり、使用する前に必ず宣言しなければなりません。宣言しないで使用した場合は翻訳の段階で未定義エラーとなります。

ここで宣言した変数はローカル変数と呼ばれ、宣言した関数内でのみ、参照／変更が可能です。ローカル変数は、関数の実行開始時に初期化され、関数の実行終了時に破壊されます。

初期化する場合、String 型の変数は、空文字列（“”）でクリアし、その他の変数は 0 でクリアします。

3.5.1. 変数の宣言

以下の形式で記述します。ローカル変数を使用しない場合は省略してもかまいません。

Dim 変数指定子１ As 変数の型 [,..., 変数指定子ｎ As 変数の型]

（例）

```
Dim x As Integer
Dim x1 As Double,x2(10) As Double,x3（10, 20, 30） As Double
```

■Dim：

予約語、変数の宣言開始を表します。

■変数指定子：

以下の形式で記述します。

（１） 単純変数の場合

変数名

（２） 配列変数の場合

以下の形式で宣言します。

Dim 変数名１（m1,..., mn）As 変数の型,..

, 変数名２（m1,..., mn）As 変数の型

m1 ：１次元目の配列要素数 - 1

mn ：ｎ次元目の配列要素数 - 1

次元数については、特に制限はありません。

（例）

```
Dim dx(10) As Integer
```

→ dx(0)から dx(9)まで 10 個の要素をもつ１次元配列の宣言

```
Dim ds(2,3) As String
```

→ ds(0,0),ds(0,1),ds(0,2),ds(0,3)

ds(1,0),ds(1,1),ds(1,2),ds(1,3)

ds(2,0),ds(2,1),ds(2,2),ds(2,3)

まで、３ｘ４個の要素をもつ２次元配列の宣言

■As：

予約語、後に変数の型が続くことを表します。

3.5.2. 変数の型宣言

変数には、以下の型があります。

型の種類	型の名前	値の範囲
ブール型	Boolean	真のとき True、偽のとき False の値を返す
文字列型	String	文字数は 0～（制限なし）
整数型	Integer	-2,147,483,648 ～ 2,147,483,647
長整数型	Long	-9,223,372,036,854,775,808 ～ 9,223,372,036,854,775,807
実数型	Double	（負の場合） $-1.79769313486232 \times 10^{308}$ ～ $-4.94065645841247 \times 10^{-324}$ （正の場合）$4.94065645841247 \times 10^{-324}$ ～ $1.79769313486232 \times 10^{308}$
エージェント種別型	AgtType	モデルツリーで定義されるエージェントの種別
エージェント型	Agt	エージェントそのもの。エージェントの実体値
エージェント集合型	AgtSet	エージェントの集合
空間型	Space	モデルツリーに定義されている空間名 ※空間のサイズは縦幅 1～10,000、横幅 1～10,000

3.6. 実行部

実行部は、ブロックと呼ばれる一連の実行文で構成され、エージェントの動作を記述します。

ブロックとは、実行文が０個以上連続したものです。

実行文には、空文、代入文、条件判断文、繰り返し文、飛び越し文等があります。

3.6.1. 空文

何も実行しない文です。プログラムをわかりやすく記述する場合に使用します。

（例）

```
If x == 1 And y == 1 Then
Else
  Z = 0
End  If
```

上記 If 文は、以下と同等です。

```
If x <> 1 Or y <> 1 Then
  z = 0
End  If
```

3.6.2. 代入文

変数に値をセットするための文です。

以下の形式で記述します。

変数指定子　＝　式

■変数指定子：

エージェントルール内で宣言した変数、または、GUI で設定したエージェント変数のいずれかです。

以下の形式で記述します。

（１）エージェントルール内で宣言した変数の場合

・ 単純変数（0 次元）の場合

変数名

(例)

x

・ 配列変数の場合

変数名

（1 次元目の要素番号を表す式,...., ｎ次元目の要素番号を表す式）

(例)

x (0)　　//１次元配列の場合

y (i)　　//１次元配列の場合

z (i,j)　　//２次元配列の場合

（２） エージェント変数の場合

エージェント指定子．変数指定子

■エージェント指定子：

当該エージェントのパス名を記述します。

(例 1) エージェントが配列の場合

Universe.agent_a（ｉ）. Agent_b

(例 2) Universe を起点にして指定する場合

Universe. Agent_a. agent_b（ｉ）

(例 3) 当該エージェントを指定する場合

My.

■変数指定子：

記述形式はエージェントルール内で宣言した変数の場合と同じです。

（代入文の例）

a = 0

a = b + Function_a（ｃ）

a (i) = b

Universe. a = 0

Universe. a (i) = b + c (i)

Universe. Agent_a（ｉ）. b = ｃ

My. a = 1

3.6.3. 式

式とは１次式、または、１次式を演算子で結合したもので、以下の種別があります。

・ 算術式 ...　　x * y + Rnd () / 10

・ 関係式 ...　　If x > y Then

・ 論理式 ...　　If x > y And x > z Then

・ 文字式 ...　　“multi_agent” = “multi_” & “agent”

１次式は、以下のいずれかです。

- **数値定数**
- **文字列定数**
- **変数指定子（代入文の変数指定子と同じ）**
- **関数呼び出し**
- **-1 次式**
- **Not 1 次式（Not：ビット演算子）**
- **（式）**

（１次式の例）

0	**定数**
x	**変数指定子**
x（i）	**変数指定子**
Function_a（a, b）	**関数呼び出し**
-x	**-1 次式**
-Function_a（a, b）	**-1 次式**
Not a	**Not 1 次式**
（a + b）	**（式）**
-（-a + b）	**-1 次式**

以下に、式の中で使用できる演算子とその優先順位をまとめました。

種別	演算子	優先順位	意味
その他	()	1	() で囲まれた式
	関数	2	関数呼び出し
算術演算子 算術式の中で使用される	^	3	べき乗、x^n
	-	4	負符号、-x
	*	5	乗算、x * y
	/	5	実数の除算（結果は実数）、x / y
	¥	6	整数の除算（結果は整数）、x ¥ y
	Mod	7	整数除算のあまり、x Mod y
	+	8	加算、x + y
	-	8	減算、x - y
文字列演算子	&	9	文字列の連結（”ab” & ”cde” →”abcde”）
関係演算子 関係式の中で使用される	= =	10	等しい（「=」(代入)との違いに注意）
	< >	10	等しくない（!= と同じ）
	! =	10	等しくない（< >と同じ）
	<	10	小さい
	< =	10	小さいか等しい

	>	10	大きい
	>=	10	大きいか等しい
論理演算子 論理式の中で 使用される	Not	10	否定
	And	11	かつ
	Or	12	または
	Xor	13	排他的論理和
代入演算子	=	14	代入（式の結果を代入する）、変数指定子＝式

3.6.4. 関数呼び出し

関数呼び出しは文法的には１次式であり、算術式、関係式、論理式の中で使用する場合と単独の文として使用する場合があります。ただし、戻り値を返さない関数として定義された関数は単独の文とする以外は使用できません。

以下の形式で記述します。

関数名（引数１，引数２...）

■引数 ：

引数の個数、各引数の型は定義した内容と一致しなければなりません。引数に指定できるのは、変数指定子だけです。

3.6.5. 条件判断文

条件によって処理を分岐させるための文です。

（１）式の値が真の場合にブロックを実行します。

（例）

```
If 式 Then
  ブロック文
End If
```

（２） 式の値が真の場合はブロック１、偽の場合はブロック２を実行します。

（例）

```
If 式 Then
  ブロック文１
Else
  ブロック文２
End If
```

（3） 式の値が真の場合に、対応するブロックを実行します。

すべての式の値が偽の場合に、ブロックｎを実行します。

（例）

```
If 式１ Then
  ブロック文１
ElseIf 式２ Then
  ブロック文２
ElseIf 式３ Then
  ブロック文３
      ・
    ・
  ・
Else
  ブロック文ｎ
End If
```

3.6.6. 繰り返し文

処理を繰り返すための文です。

（1） For 文

処理を所定回数繰り返す場合に使用する。以下の形式で記述します。

For ループ変数＝初期値 To 最終値 (Step きざみ値) ※省略時きざみ値１

ブロック文

Next ループ変数

(例)

```
total = 0
For i=1 To 10 Step 1
    total = total + i
Next i
```

エージェント集合の各エージェントに対して同じ操作を行う場合は、以下の形式も利用できます。

For Each ループエージェント In エージェント集合

ブロック文

Next ループエージェント

(例)

```
For Each TmpAgt In カメ集合
  TmpAgt.X = 25.0
Next TmpAgt
```

（2） Do While 文

式の値が真の間繰り返す場合に使用します。

以下の形式で記述します。

Do While 式

ブロック文

Loop

（3）Do Until 文

式の値が偽の間（真になるまで）繰り返す場合に使用します。

Do Until 式

ブロック文

Loop

3.6.7. 飛び越し文

制御の流れを変更するための文です。ただし、多用するとプログラムの構成が分かりにくくなるため、なるべく利用は避けるようにしてください。

（1） Goto 文

指定した場所に飛び越す場合に使用する。以下の形式で記述します。

Goto 飛び越し先ラベル名

・

・

・

飛び越し先ラベル名：

・

・

・

（例）

```
Do  While  x  >  0
  ・・・
  If err_flag == True Then
    Goto error_syori              指定したラベルに飛び越す
  End If
Loop
error_syori:
  ・・・
```

（2） Break 文

途中でループから抜ける場合に使用します。

（例） While 文のループを途中で抜ける場合

```
Do While 条件式
  .
  .
If  脱出条件式  Then
    Break                                →一つ外に脱出する
End If
Loop
```

(例)　For 文のループを途中で抜ける場合

```
For i=0 To 10 Step 1
  For j=0 To 10 Step 1
    .
    .
    If  脱出条件式  Then
      Break     →一つ外に脱出する
    End If
  Next j
  .
Next i
```

3.6.8.　　インクルード文

外部ファイル（以下、インクルードファイルとする）に定義した関数を読み込むための文です。よく利用する関数をインクルードファイルに記述することで、複数のエージェントから同一の関数を呼ぶことができます。通常、ルールエディタの先頭行に以下の形式で記述します。

include “インクルードファイル名１.inc”

include “インクルードファイル名２.inc”

＜注意＞：include 文は、関数内でいくつ記述しても構いません。

＜注意＞：include 文を複数定義する場合は、１ファイル１行で記述してください。

＜注意＞：Universe で定義した関数および Universe でインクルードファイル
として定義された関数は、エージェントから「@関数名」で呼び出す
ことができます。

(例)　include 文を定義する

```
include “インクルードファイル名１.inc”
include “インクルードファイル名２.inc”

Univ_Init{
  .
  .
  .
```

```
}
```

3.6.9. その他

（1） Return 文

関数の実行を終了して、呼び出し側に実行の制御を戻すための文です。このとき、関数が戻りを返す関数として定義してあれば値を返すことができます。以下の形式で記述します。

```
Return （式）
```

＜注意＞：Return 文は、関数内で、いくつ記述しても構いません。

＜注意＞：以下の場合は翻訳エラーとなってしまいます。

- 値を返す関数として定義して、式を省略した場合
- 値を返す関数として定義して、戻り値の型と式の型が異なる場合
- 値を返さない関数として定義して、式を記述した場合

(例) Return 文を定義する

```
Function abc() As Integer
{
  Dim z As Integer
  .
  .
  Return(z)
}
```

3.7. 組込み関数

マルチエージェントシミュレータでは、組込み関数と呼ばれる予め定義済の関数があります。これらの関数はすべての関数内から呼び出すことができます。

組み込み関数の詳細につきましては、artisoc ヘルプを参照下さい。

3.8. 予約語

エージェントルールを記述する際、変数名、ユーザ関数名等に予約語を使用することはできません。また、組込み関数の名称も使用することはできません。

予約語の詳細につきましては、artisoc ヘルプを参照下さい。

3.9. コメント

エージェントルールにはコメントを記述することができます。コメントには何を書き込んでも構いません（artisoc はそのコメントの部分を無視するからです)。ルール文は人間が即座に理解しにくい英文の羅列になり易いので、後で見返す時のためにも、なるべく多くのコメントを入れておくことをお勧めします。なお、コメントは空白が記述できるところならばどこに記述しても構いません。

以下にコメントの記述方法を示します。

（1） ’

' から行の終わりまでがコメントとなります。

（この場合、『sum = 0』の部分までが、ルールとして翻訳されます）

(例)

```
sum = 0   ’合計値の初期化
```

（2） //

（1） と同様

（例）

```
//  自分の進む方向を決定して移動する。
direction = Rnd (360)
```

（3） /* ... */

/* から */ までがコメントとなります。(複数行にまたがっても構いません)

(例)

```
/* ***********************************************
* カメ　ルール                                  *
* シェリングの分居モデルを考察する              *
***********************************************/
```

3. 10. エラーの対処法

ルールエディタにて文法的に誤ったルールを記述すると、ウィンドウ下部のステータスバーにエラーが表示されます。エラー表示をヒントに正しいルールを記述してください。

また、デバッグ機能を利用すると 1 行ずつ実行することができます。

デバッグ中の変数の値は、ステップイン、ステップオーバーを実行している間にデバッグ画面で Print、PrintLn 関数を記述し、Return キーを入力するとコンソール画面にその値が表示できます。

例えば、デバッグ中の「Universe.abc」の変数値の値を表示したい場合はデバッグ画面で

```
PrintLn(Universe.abc)
```

とすることによりコンソール画面で値を確認できます。
また、

```
Universe.abc = 100
```

とすると変数への代入ができます。

さらに、出力設定の値画面出力にて任意の変数値を定義し、デバッグすると値を確認できます。