

11. 簡単な交通流モデル

2017/12/18

1 はじめに

前回の実験で扱った Wolfram の CA のうち、Rule-184 は、最も簡単な交通流のモデルとなります。。交通流のモデルとして必要な要素は

- 排除体積効果：一つの場所を複数の車両が占めることができない。
- 挙動の遅れ：前の車の挙動に対して、挙動が遅れる。

と考えられます。Rule-184 で、セルの値が 1 であることをそのセルに粒子が居る、セルの値が 0 であることをそのセルに粒子が居ないと解釈することにしましょう。すると、交通流モデルの基本的要素は、Rule-184 において

- 一つのセルは一つの粒子しか占めることができない。
- 前のセルが空かないと、前進できない。

という形で表現されます。今回は、Rule-184 の性質を、交通流のモデルという観点から調べましょう。

課題 1 `processing` を用いた動画において、ルール番号を 184 とするとともに、初期化を `initialSingle()` メソッドから `initialize(double r)` へ変更しましょう。ここで r は車両の初期密度に対応します。 $r < 0.5$ と $r > 0.5$ の場合の挙動の違いを観察し、記述しましょう。

2 Observable クラス

次に、交通流モデルとして見たときの観測量を調べるために、以下の URL から Java ソースファイルを取得し、プロジェクト CA のソースファイルの下に置きます。

<http://http://aoba.cc.saga-u.ac.jp/lecture/ModelingAndSimulation/javasrc/CA/src-rule184.zip>

周期境界条件の下での Rule-184 では、値が 1 であるセルの総数は変化しません。交通流に当てはめると、車両の数は変化しません。その数を N とします。車両密度は、セルの総数を L として、 $\rho = N/L$ で定義します。

交通流としての性質を調べるために、密度を変えながら、平均速度と流量などの観測量を調べることにします。そこで、共通的な部分を抽象クラス `Observable` に定義しておきます。

表 1 `Observable` クラスのコンストラクタ

コンストラクタと説明
<code>Observable(int n)</code> サイト数 n を与えて、rule-184 の CA を初期化する。

表 2 `Observable` クラスのメソッド

修飾子と型	メソッドと説明
<code>abstract double</code>	<code>calcValue(double p, int tmax)</code> 密度 p に対して、平均時間 $tmax$ で観測量を求める抽象メソッド。
<code>List<Point2D.Double></code>	<code>calcValues(double dp, int tmax)</code> 密度を dp ずつ変化させながら、平均時間 $tmax$ で観測量を求める。
<code>protected void</code>	<code>initializeAndRelax(double p, int tmax)</code> 観測量を実際に計測するまえに、システムを初期化し、密度 p に対して時間 $tmax$ だけ緩和させる。

3 平均速度

第一の観測量として平均速度を定義します。ここで、「平均」とは空間と時間に関する平均と考えることにします。「空間的平均」とは、ある時刻ですべての車両に関する平均をとることと定義します。この空間的平均を、さらに時間について平均をとることとします。

速度の空間的平均は、各車両の速度を加算して、車両数で除することで定義できます。しかし、Rule-184 で表される交通流モデルでは、車両の速度は 0 か 1 です。つまり、「各車両の速度を加算」とは、動いた車両の数を数えることと等価です。

セル i の時刻 t での値を $S_i(t)$ と表すことにします。時刻 t にセル i に居る車両が動くということは、

$$S_i(t) = 1, \quad S_{i+1}(t) = 0 \quad (3.1)$$

という状態から

$$S_i(t+1) = 0, \quad S_{i+1}(t+1) = 1 \quad (3.2)$$

となるということです。つまり、値が変化しているセルの数は、動いた車両の 2 倍になります。時刻 t にセル i に居る車両が動かない場合には、セル i の値は変化しません。

クラス `CA` の更新メソッド `update` は更新後のセルの値の配列を戻り値として返すだけでなく、値が更新されたセルの数 `numDifference` を計算しています。この値を得るメソッドは `getNumDifference()` です。

クラス `Speed` は、Rule-184 クラスのシミュレーションを、車両密度を変えながら実行し、車両密度に対する平均速度を求めるための、`Observable` クラスの拡張クラスです。

表 3 `Speed` クラスのコンストラクタ

コンストラクタと説明
<code>Speed(int n)</code> サイト数 n を与えて、rule-184 の CA を初期化する。

課題 2 メソッド `public double calcValue(double p, int tmax)` は、車両密度 p と、時間平均をとる時間 $tmax$ を与え、平均速度を戻り値として返します。そのために、最初に `initializeAndRelax()` を用いて、初期化と緩和を行います。その後に、移動した車両の数を $tmax$ 回求めて加算し、最後に $tmax$ 及び車両の数で除した値を返します。このメソッドを完成させましょう。

課題3 また、クラス `rule184.Speed` を実行し、データを曲線の式

$$y = \begin{cases} 1 & x < 1/2 \\ \frac{1}{x} - 1 & \text{otherwise} \end{cases} \quad (3.3)$$

とともに描きましょう。。なお、gnuplot で場合分けのある式を定義するには、C/C++ と同様に

`f(x) = 条件式 ? 式1 : 式2`

と書きます。

4 流量

交通流の研究では、密度に対して流量がどのように変化するかを調べることも重要です。流量とは、ある場所を単位時間に何台の車両が通り抜けたかを表します。

課題4 クラス `rule184.Flow` は、Rule-184 クラスのシミュレーションを、車両密度を変えながら実行しする、`Observable` クラスの拡張クラスです。流量を求める `calcValue()` メソッドを完成させましょう。流量は、`tmax` の間に移動した車両数を、時間 `tmax` 及びセルの総数で除した値です。クラス `CA` には、セルの総数を得るメソッド `getN()` があります。

表4 Flow クラスのコンストラクタ

コンストラクタと説明
<code>Flow(int n)</code> サイト数 <code>n</code> を与えて、rule-184 の CA を初期化する。

課題5 また、クラス `rule184.Flow` を実行し、データを曲線の式

$$y = \begin{cases} x & x < 1/2 \\ 1 - x & \text{otherwise} \end{cases} \quad (4.1)$$

とともに描きましょう。