セルオートマトン

モデリングとシミュレーション

2016年度

離散化の可能性

- 離散(discrete)
  - 値が飛び飛び、例えば整数の値しかとらない
- 連続的な時間ではなく、ある時間間隔で観測する
- 空間もある間隔で観測する
- 空間や状態を離散化

離散化の利点・欠点

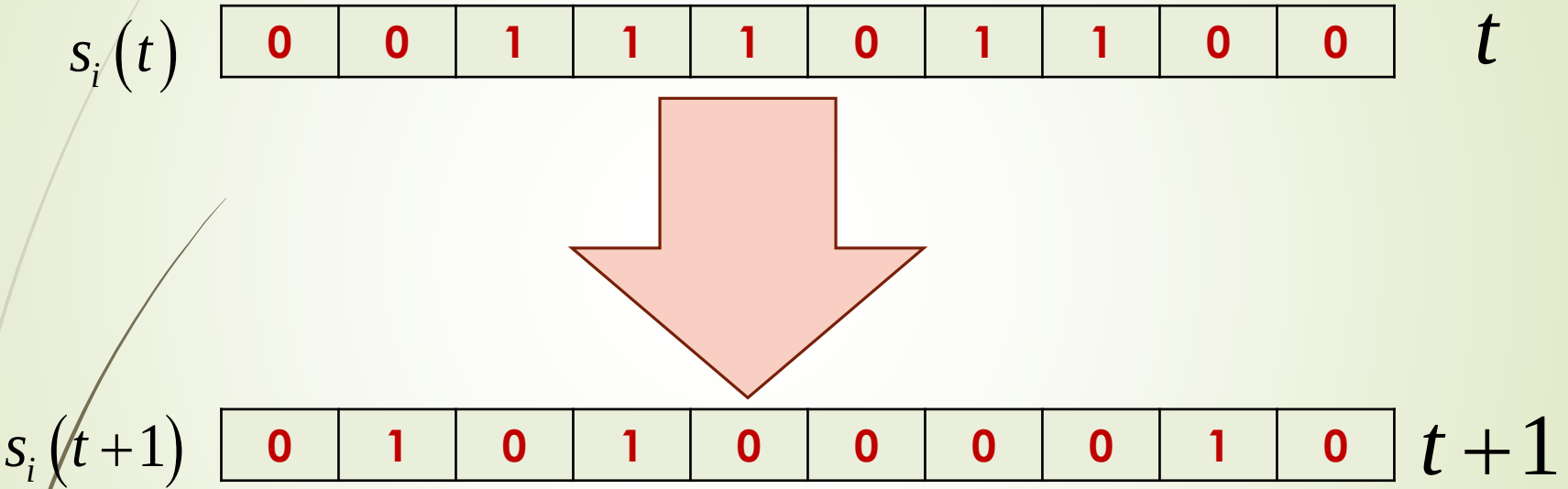
- 微分方程式に書けない場合
  - 発展の規則として記述できる
  - 本当に正しいかの検証が必要
- シミュレーション
  - 規則として記述できる
  - 整数演算は高速
  - 計算誤差が出ない

セルオートマトン Cellular Automata

- 空間をセル(ます)に区切る
- 時間を離散化する
- 時間発展規則を定める
  - 周囲の状態で次の状態を定める
- automata
  - automaton の複数形

1次元セルオートマトンの例

$$s_i(t+1) = \left(s_{i-1}(t) + s_i(t) + s_{i+1}(t)\right) \bmod 2$$



1次元セルオートマトン

- 隣接する$2r+1$個のセルの状態で、次の時刻の状態を決定

$$s_i(t+1) = F\left(s_{i-r}(t-1), s_{i-r+1}(t-1), \cdots, s_i(t-1), \cdots, s_{i+r}(t-1)\right)$$

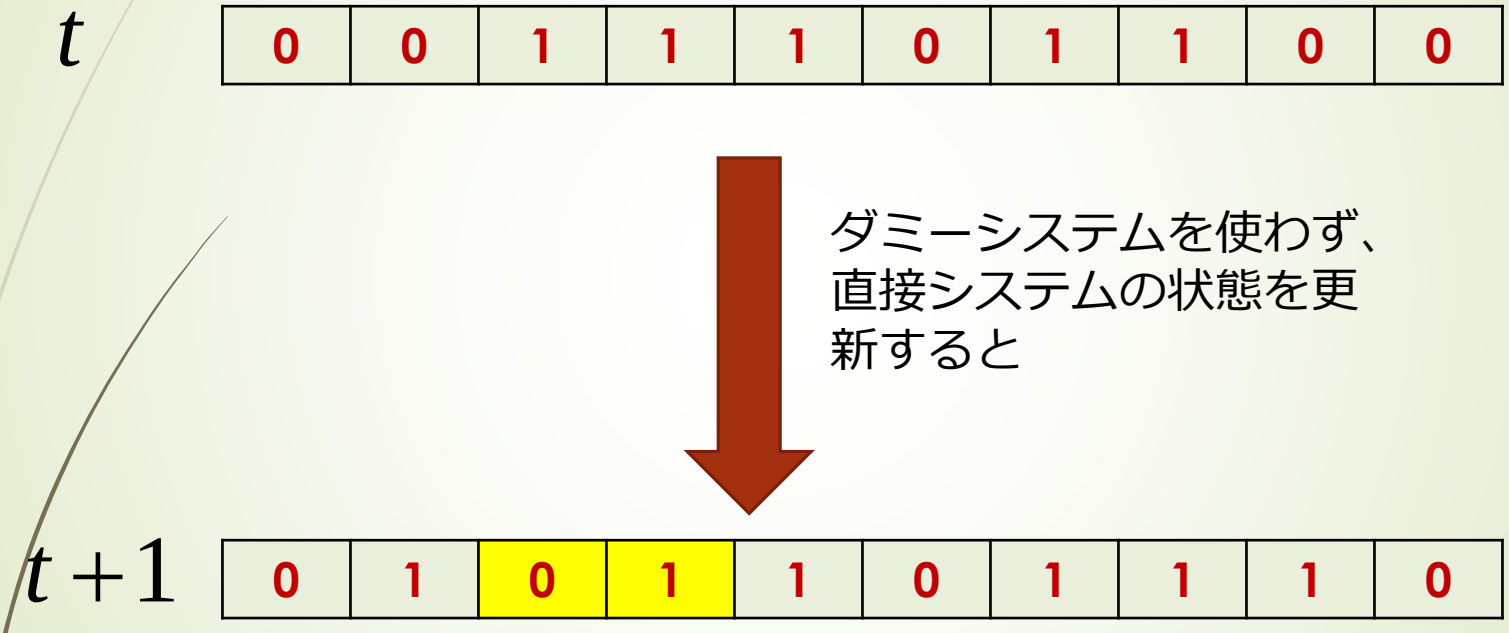
- すべてのセルに一つの規則を適用
- すべてのセルを同時に更新
  - コンピュータは、すべての状態を同時に更新できない。どうする？

同期的更新：一般論

- 複数のセルなどの状態を、同期的（つまり同時に）更新するには
  - コンピュータはそんなことはできない
- 次の時刻の状態を入れる専用のデータ構造（ダミーと呼ぶ）を作り、そこに書き込む
  - 元のシステムの状態は更新しない
- ダミーの状態を、システムに書き込む

同期的更新の例

左端のセルから更新すると



最も簡単な1次元セルオートマトン

- 内部状態$\{0,1\}$
- $r = 1$

$$s_i(t+1) = F\left(s_{i-1}(t-1), s_i(t-1), s_{i+1}(t-1)\right)$$

- 右辺の引数のパターンは3bit=8通り
- 8通りの入力に0か1を割り当てる
  - $2^8 = 256$通り

入力	111	110	101	100	011	010	001	000
出力	1	0	1	1	1	0	0	0

$$(10111000)_2 = 184$$

入力	111	110	101	100	011	010	001	000
出力	0	1	0	1	1	0	1	0

$$(01011010)_2 = 90$$

Wolframの基本CA：左右対称なもの

実際にやってみよう

Rule-90 周期境界条件

0	**0**	**0**	**0**	**1**	**0**	**0**	**0**	**0**

実際にやってみよう

Rule-184 周期境界条件

1	0	1	1	1	0	1	0	1

Ruleから作ろう

- Rule 150

入力	111	110	101	100	011	010	001	000
出力								

- Rule 54

入力	111	110	101	100	011	010	001	000
出力								

実際にやってみよう

Rule-150 周期境界条件

0	**0**	**0**	**1**	**0**	**1**	**0**	**0**	**0**

実際にやってみよう

Rule-54 周期境界条件

1	0	1	1	1	0	1	0	1

- Rule 90

入力	111	110	101	100	011	010	001	000
出力	0	1	0	1	1	0	1	0

$$(01011010)_2 = 90$$

実際にやってみよう

Rule-90 周期境界条件

t									
	1	**0**	**1**	**1**	**1**	**0**	**1**	**0**	**1**