

# Intelligenza Artificiale II

## Automati cellulari Introduzione

Marco Piastra

## ***Self-Organization*** e calcolo distribuito

- Calcolo complessivo, unico flusso di calcolo

Esempi: il *k-means* (algoritmo di Lloyd) l'algoritmo *EM*

Ad ogni iterazione vengono considerati tutti i dati in  $D$

Il calcolo non è distribuito

(anche se possono essere parallelizzate le operazioni, p.es. le sommatorie)

- Calcolo locale, sostanzialmente distribuito

Esempi: *Hard Competitive Learning*, *Self-Organizing Map*, *Growing Neural Gas*

Il calcolo avviene in modo incrementale, con segnali che arrivano in sequenza

Ciascun segnale ha effetto su un'area limitata della struttura

Il calcolo può essere effettuato in modo distribuito

(con qualche accorgimento)

- Calcolo locale, intrinsecamente distribuito

Automati cellulari

# Game of Life (J.H. Conway, 1970)

- **Universo**

Reticolo ortogonale infinito di celle quadrate

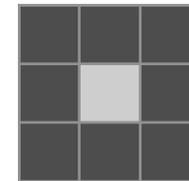
Le celle sono tutte uguali (universo omogeneo)

- **Stati**

Ciascuna cella può essere viva o morta (1 o 0)

- **Intorno (di una cella)**

Ciascuna cella ha un intorno formato dalle otto celle contigue



- **Dinamica**

L'universo si evolve nel tempo, a partire da una configurazione iniziale

Evoluzione parallela:

Ciascuna cella decide il proprio stato futuro in base al proprio stato ed allo stato delle celle nel suo intorno

Tempo discreto:

Ad ogni istante, ciascuna cella dell'universo decide quale stato avrà nell'istante successivo

# Game of Life (J.H. Conway, 1970)

- **Universo**

Reticolo ortogonale infinito di celle quadrate

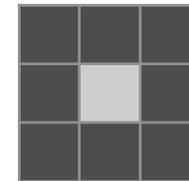
Le celle sono tutte uguali (universo omogeneo)

- **Stati**

Ciascuna cella può essere viva o morta (1 o 0)

- **Intorno (di una cella)**

Ciascuna cella ha un intorno formato dalle otto celle contigue



- **Dinamica**

Regole di transizione:

- 1) Una cella viva muore se l'intorno contiene meno di due celle vive (*solitudine*)
- 2) Una cella viva muore se l'intorno contiene più di tre celle vive (*sovrappopolazione*)
- 3) Una cella viva rimane viva se l'intorno contiene due o tre celle vive (*mutuo sostegno*)
- 4) Una cella morta con tre celle vive nel suo intorno diventa viva (*genesì*)

# Game of Life: configurazioni e dinamica

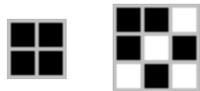
*L'aspetto emergente è il comportamento della popolazione di celle vive*

- Configurazioni iniziali

Il comportamento è completamente determinato dalla configurazione iniziale

- Comportamenti possibili (vedi dimostrazione con Golly)

Nessuna variazione



Comportamento periodico



Altri comportamenti ....



# Game of Life: configurazioni e dinamica

*L'aspetto emergente è il comportamento della popolazione di celle vive*

- Configurazioni iniziali

Il comportamento è completamente determinato dalla configurazione iniziale

- Altre categorie (informali)

*Still lifes*

Pattern statici o quasi statici

*Oscillators*

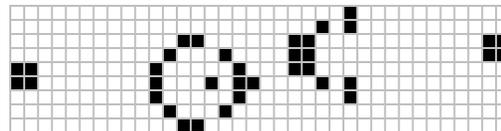
Pattern che si riproducono dopo un certo periodo di tempo

*Spaceships*

Pattern che si 'spostano' nell'universo

*Guns*

Pattern che 'emettono' *spaceships*



# Game of Life: configurazioni e dinamica

*L'aspetto emergente è il comportamento della popolazione di celle vive*

- **Configurazioni iniziali**

Il comportamento è completamente determinato dalla configurazione iniziale

- **Macchine complesse**

*Glider gun come componenti logici*

Normalmente i *glider gun* 'emettono' segnali (stato *true*)

Possono essere accoppiati con un *eater* che inibisce l'emissione (stato *false*)

*Turing machine*

Combinando opportunamente tali componenti,  
si può costruire una macchina da calcolo universale  
(vedi esempio in Golly)

# Classi di automi cellulari

*Classificazione basata sul comportamento, non sulla struttura o le regole*

- **Classe 1: attrattore singolo**

Dopo un numero finito di passi, l'automata tende ad un'unica configurazione stabile, a partire da quasi tutte le configurazioni iniziali

- **Classe 2: ciclo limite**

L'automata produce schemi che si ripetono periodicamente, all'infinito (probabilmente equivalente alla classe degli automi a stati finiti)

- **Classe 3: caotico**

L'automata produce schemi aperiodici e/o caotici (assomigliano a rumor bianco), che hanno proprietà statistiche quasi identiche alle configurazioni iniziali, almeno dopo un certo numero di passi (configurazioni auto-simili).

- **Classe 4: strutturato**

L'automata produce schemi complessi che si muovono nello spazio, nel tempo. Nella classe 4 si trovano automi equivalenti alla *Universal Turing Machine*

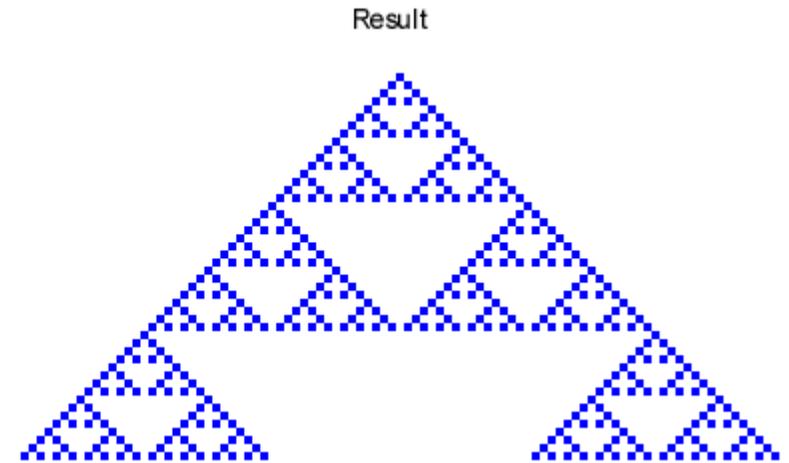
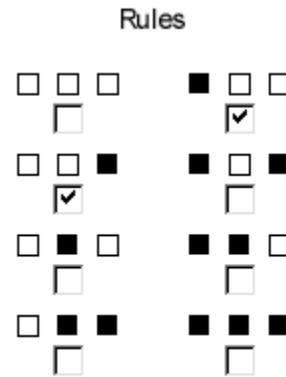
# Automati unidimensionali

- Una sola riga di celle

Ad ogni step, tutte le celle, simultaneamente cambiano stato

Le transizioni sono guidate da regole: il nuovo stato dallo stato della cella e delle celle vicine

Forma più semplice: tutte le celle dello stesso tipo, due soli stati possibili, la transizione dipende solo dallo stato delle celle immediatamente contigue



Regole di transizione (esempio)

Stato precedente	1	1	1	1	1	0	1	0	1	1	0	0	0	1	1	0	1	0	0	0	1	0	0	0
Nuovo stato	1			0			1			1			0			1			1			1		

Assumendo un ordinamento standard, ciascuna regola è descritta da otto bit

Nell'esempio è 10110111 (Regola 183)

L'automata è formato da una sola riga di dimensione finita, ciclica (la prima e l'ultima cella sono adiacenti)

Lo stato iniziale della riga è l'unico input

La traccia dell'esecuzione è formata dalla lista degli stati successivi della riga

# Esempi

(usare l'applet al link: <http://www.alesdar.org/oldSite/IS/OneDCA/OneDCA.html>)

- Classe 1

Regola 32: 00100000

Regola 160: 10100000

- Classe 2

Regola 33: 00100001

- Classe 3

Regola 126: 01111110

Regola 30: 00011110

- Classe 4

Regola 110: 01101110

E' una macchina di Turing

(vedi [http://en.wikipedia.org/wiki/Rule\\_110](http://en.wikipedia.org/wiki/Rule_110))