Life: un mondo artificiale brulicante di imprevedibili organismi viventi

Giorgio Meini

Un sistema relativamente complesso e rigidamente deterministico di cui si conosce senza errore la legge che ne regola il comportamento e lo stato iniziale può non essere prevedibile nella sua evoluzione temporale fino al punto da limitare all'osservazione i possibili strumenti d analisi

Ma c'è un aspetto del gioco per cui il suo nome "Life" non è giustificato. Il gioco mostra molto bene come da semplici regole di base si possa costruire un "mondo reale" estremamente complesso, e complesso non solo nella sua struttura spaziale, nella sua morfologia, ma anzitutto nel suo comportamento funzionale. Troviamo qui di fatto molte proprietà che richiamano il concetto di vita, in senso sia positivo che negativo. Manca però una caratteristica essenziale della vita: il caso "creativo". Il decorso del gioco "Life" è esemplare di un processo deterministico. Tutto procede in modo strettamente causale a partire dalle condizioni iniziali. Dietro a tutto si cela un "creatore pienamente informato". Il gioco si limita a rivelare le sue idee. Il caso, semplicemente, non è previsto.

[M. Eigen, R. Winkler, *Il Gioco*, trad. it. Adelphi ed., 1986]

Nel 1970, dalle pagine di *Scientific American*, Martin Gardner fece conoscere al mondo intero un gioco - denominato *Life* - ideato dal matematico inglese John H. Conway allo scopo di simulare lo sviluppo e il decadimento di popolazioni composte da semplici organismi viventi. Nel gioco lo spazio è rappresentato da una griglia bidimensionale di caselle (avente idealmente estensione infinita) e ognuna delle quali può ospitare, o meno, un singolo organismo. Dovendo limitare l'estensione dello spazio ed evitare al tempo stesso di avere caselle "di confine" è possibile considerare adiacenti le caselle dei lati opposti della griglia: la topologia dello spazio di gioco diviene così quella di un toro (**figura 1**).

Il tempo è una successione indefinita di istanti discreti nel trascorrere dei quali la popolazione si modifica secondo queste semplici regole:

- soppravvivenza: un organismo sopravvive se nelle caselle adiacenti sono presenti 2 o 3 organismi;
- morte: un organismo muore liberando la casella che occupa se nelle caselle adiacenti si trovano più di 3 (sovraffollamento) o meno di 2 (isolamento) organismi;
- nascita: una casella vuota viene occupata da un nuovo organismo se nelle caselle adiacenti sono presenti esattamente 3 organismi.

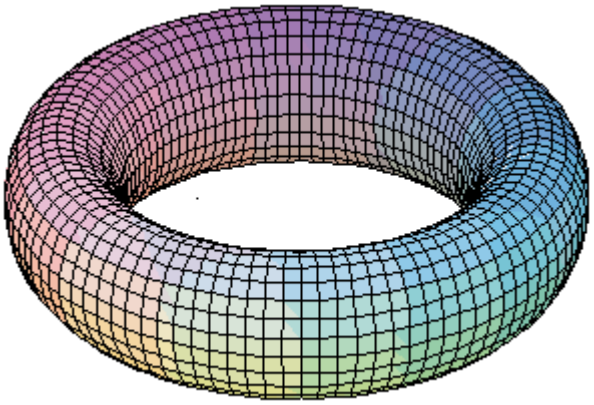

Figura 1 - Disposizione toroidale delle celle nello spazio del gioco.

Per caselle adiacenti di una particolare casella si intendono le 8 che ne costituiscono l'intorno per prossimità ortogonale e diagonale. Il gioco consiste nel prevedere ed osservare l'evoluzione nel tempo di una popolazione iniziale di organismi variamente distribuita nello spazio. Esistono disposizioni che evolvono verso strutture spaziali stabili nel tempo - statiche o periodiche - e disposizioni instabili che si accrescono indefinitamente o che, dopo una fase transitoria più o meno duratura, si estinguono completamente (ed è questo il caso più comune). Prima di affrontare la parte più interessante della storia invito i lettori a sperimentare almeno la "dinamica" delle disposizioni iniziali di organismi della **figura 2**.

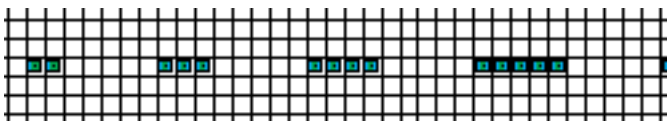

Figura 2 - Cinque configurazioni iniziali aventi una semplice dinamica evolutiva.

Computer Programming n. 40, Ottobre 1995, pp. 46-52

L'evoluzione nel tempo di queste strutture minimali non è certo entusiasmante ed è ancora facilmente prevedibile, ma da sole rappresentano comunque 3 dei 4 esiti plausibili: estinzione e stabilità statica o periodica dopo un transitorio iniziale. Certamente il fascino del gioco consiste nell'osservare la dinamica di disposizioni molto più complesse, eventualmente ottenute dalla contaminazione per prossimità spaziale di strutture prodotte da semplici configurazioni iniziali aventi orientamento diverso (si noti che le regole del gioco sono simmetriche rispetto agli orientamenti ortogonali e diagonali e che questa caratteristica di "isotropia" favorisce l'evoluzione di aggregazioni di organismi simmetriche e super-simmetriche). Sicuramente il futuro di una particolare disposizione di organismi è univocamente determinato dalle regole del gioco, ma è sempre possibile prevedere a priori l'esito di una arbitraria configurazione iniziale? In altre parole: è possibile, almeno in linea di principio, scrivere un programma che, data una specifica disposizione (*pattern*) di organismi, determini in un tempo quantificabile l'esito finale della loro evoluzione secondo le regole del gioco? Vedremo che la risposta è negativa: *Life* è veramente imprevedibile! Questa domanda e questa risposta hanno attinenza con le cosiddette "teorie del tutto": chi è interessato all'argomento può divertirsi a leggere il breve racconto matematico di Ian Stewart intitolato "La Risposta definitiva" e dedicato ad un sistema matematico - la formica di Langton - ideato dal ricercatore della *Artificial Life* Chris Langton, sul numero 313 (Settembre 1994) della rivista Le Scienze.
Esiste una configurazione - e non è la sola - che non si estingue, non si accresce indefinitamente, non si stabilizza né staticamente né periodicamente: è quella rappresentata nella **figura 3**.

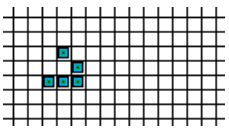

Figura 3 - La configurazione denominata "aliante".

Si tratta in effetti di un *pattern* - denominato *glider*, aliante - periodico rispetto al tempo, ma non rispetto allo spazio: infatti esso si muove nella griglia del gioco con una velocità pari ad un quarto della velocità massima teorica (la "velocità della luce" è ovviamente pari a quella del Re nel gioco degli Scacchi). Considerando uno spazio di gioco finito come il toro una popolazione che non si estingua non può che stabilizzarsi in una configurazione statica o periodica: il numero di configurazioni possibili è infatti grande, ma non infinito (per una griglia di m X n caselle le configurazioni possibili sono esattamente 2^{mn}) e, di conseguenza, non può esistere una popolazione che varia eternamente la propria disposizione: prima o poi (al massimo dopo 2^{mn} "quanti" di tempo) si disporrà secondo una configurazione già assunta in precedenza e, dato che le regole sono invarianti rispetto al tempo, a partire da quell'istante si stabilizzerà in un ciclo periodico (questo ragionamento è un esempio di ciò che in Matematica si definisce dimostrazione formale: è sempre necessario ricorrervi per evitare di verificare direttamente un numero proibitivo o infinito di casi particolari). Passando ad un ipotetico spazio di gioco infinito si aprono nuove possibilità:

- strutture periodiche nel tempo rispetto allo spazio locale, ma non a quello globale (è il caso del movimento, di cui il *glider* costituisce un semplice esempio),
- popolazioni il cui numero di organismi cresce indefinitamente (con o senza una dinamica prevedibile),
- popolazioni che modificano continuamente la propria disposizione mantenendosi limitate e non conservando una struttura periodica locale (anche in questo caso la dinamica può essere o non essere prevedibile).

Conway stesso non riuscì a determinare una disposizione iniziale per una popolazione il cui numero di organismi aumentasse continuamente e offrì un premio di 50 dollari al primo che vi fosse riuscito o che avesse dimostrato l'impossibilità di un accrescimento infinito. L'offerta venne pubblicizzata da Martin Gardner nella propria rubrica mensile su *Scientific American* e poco tempo dopo un piccolo gruppo di giovani ricercatori del gruppo di Intelligenza Artificiale del MIT di cui faceva parte Bill Gosper propose il *glider gun* della **figura 4**.

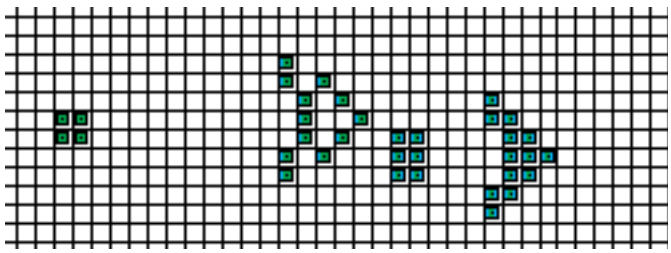

Figura 4 - Il "cannone di alianti".

Il "cannone di alianti" genera un nuovo *glider* ogni 30 unità discrete di tempo (è questa la durata del suo ciclo di oscillazione): in uno spazio illimitato gli alianti si mantengono indefinitamente in movimento e quindi il numero complessivo di organismi si accresce continuamente. Esiste anche una struttura periodica capace di "mangiare" gli alianti che la incontrano in determinati istanti del proprio ciclo (avente durata di 15 quanti di tempo) e con una particolare direzione di incidenza: chi dei lettori è in grado di trovarla?
Personalmente non conosco, pur non escludendo che esista, una disposizione spaziale di organismi che generi - in uno spazio illimitato - una popolazione che si mantenga limitata come dimensione, ma che modifichi in modo sicuramente non periodico - anche rispetto ad un riferimento spaziale locale come avviene nel caso del *glider* - la propria configurazione. Chi dei lettori sia in grado di trovare una tale configurazione iniziale è invitato ad inviarla alla redazione della rivista: sarà ricompensato con ... una cita-

zione personalizzata in uno dei prossimi numeri! La ricerca può senz'altro iniziare con alcuni esperimenti esplorartivi condotti con il *computer*, ma - dato che, come abbiamo visto, l'esito ricercato è possibile solo in uno spazio infinito - la certezza "matematica" è raggiungibile solo affidandosi ad un ragionamento formalizzato.

Per quanto la simulazione visuale al *computer* del gioco *Life* comunichi l'impressione di vitalità propria di una colonia di organismi, non sono poi molti i "comandamenti" della *Artificial Life* che risultano, almeno a prima vista, soddisfatti. In questo senso una importante domanda da porsi è la seguente: - Esistono strutture di organismi in grado di auto-riprodursi generando nello spazio contiguo una seconda struttura avente identica morfologia (disposizione relativa dei singoli elementi della configurazione)?
La risposta è: - Teoricamente sì. Ma per motivare questa affermazione dobbiamo tornare indietro almeno fino ai primi anni '50. *Life* è un esempio di una classe di sistemi computazionali astratti - oggi denominati Automi Cellulari - ideati alla fine degli anni '40 dai matematici John von Neumann e Stanislaw Ulam. Un automa cellulare è caratterizzato da un insieme di semplici elementi di computazione identici tra loro e con una geometria delle relazioni reciproche deteminata a priori. Ogni singola "cellula" può ad ogni istante di un tempo che fluisce per quanti discreti assumere un particolare stato - il numero degli stati possibili può variare, ma è lo stesso per tutte le cellule - determinato dal suo stato precedente e dagli stati delle cellule considerate vicine secondo una topologia propria dello specifico automa. Von Neumann aveva dimostrato in precedenza che una "macchina" (astratta) poteva replicare sé stessa per mezzo di un "costruttore universale", un componente computazionalmente equivalente ad una macchina di Turing universale(1).

Poco prima di morire - i risultati della ricerca sono stati pubblicati in seguito da Arthur Burks - riuscì a dimostrare che un automa cellulare composto da circa 200.000 cellule, ognuna delle quali avente 29 stati distinti conteneva un costruttore universale: fino ad oggi nessuno ha mai simulato con un *computer* l'automa di von Neumann! E' stato dimostrato - accenneremo nel seguito al metodo seguito - che nel gioco *Life* esistono configurazioni computazionalmente equivalenti alla macchina di Turing: in linea di principio è quindi possibile simulare l'automa cellulare di von Neumann che auto-riproduce la propria struttura.

1 Una macchina di Turing universale è un modello astratto di computazione in grado di eseguire qualsiasi algoritmo opportunamente codificato: rappresenta il concetto matematico di calcolabilità. Una macchina di Turing universale è formalmente equivalente ad un calcolatore convenzionale - la cui struttura fondamentale è stata definita da von Neumann negli anno '40 - avente una memoria potenzialmente illimitata.

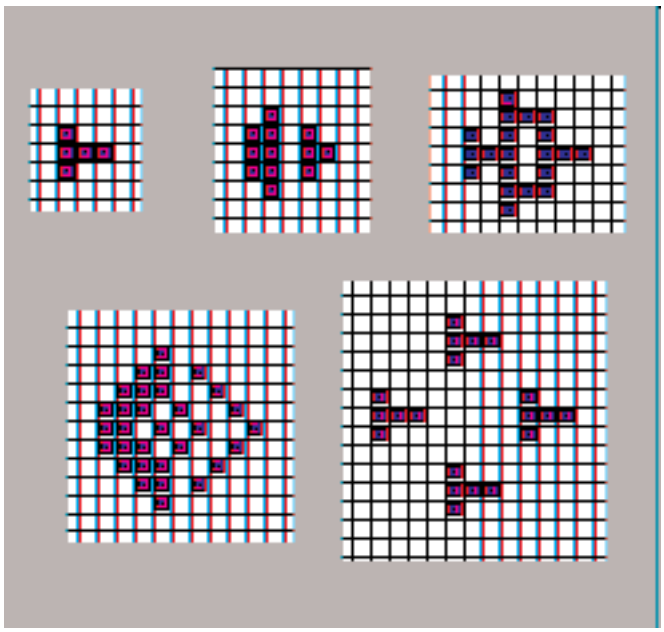

Figura 5 - Struttura che si autoriproduce nel gioco di Fredkin.

La **figura 5** illustra una semplice struttura che si auto-riproduce evolvendosi in un gioco - ideato negli anni '60 da Edward Fredkin - simile a *Life*, ma con regole di nascita, di sopravvivenza e di morte diverse. In questo caso lo stato successivo di una casella è determinato esclusivamente dallo stato presente delle quattro caselle adiacenti ortogonali (non sono considerate le caselle adiacenti diagonali): se il numero totale di organismi presenti nell'intorno è pari (2 o 4) la casella non ospiterà un organismo (se già lo ospita muore), se è dispari (1 o 3) la casella sarà abitata da un organismo (se è attualmente vuota nasce). Esiste una struttura relativamente semplice ed avente una simile proprietà anche per il gioco *Life*? Non ne sono a conoscenza, ma forse qualcuno dei miei "venticinque lettori" può fornire un contributo in questo senso (limitandosi ad uno spazio finito è teoricamente possibile - trascurando cioè l'intrascurabile aspetto del tempo di computazione - condurre una ricerca di questo tipo in modo automatico, per mezzo di un programma che osservi per un adeguato numero di "generazioni" l'evoluzione di molte disposizioni iniziali diverse).

Come è possibile che un sistema semplice come il gioco *Life* sia equivalente al modello di calcolo universale rappresentato dalla macchina di Turing? Ricordate il cannone di alianti? Bene, il "fascio" di alianti ottenuto rappresenta un segnale di sincronismo analogo al *clock* di un calcolatore digitale: un fascio sincronizzato con questo segnale di *clock* può rappresentare una informazione binaria se la presenza di un aliante viene interpretata come "1" e l'assenza come "0".
Nella **figura 6** il fascio a sinistra rappresenta il segnale di sincronismo, mentre il fascio a destra è la codifica dei bit 1-0-1; inoltre i due fasci sono reciprocamente orientati in modo da costituire una porta NOT (provare per credere: il

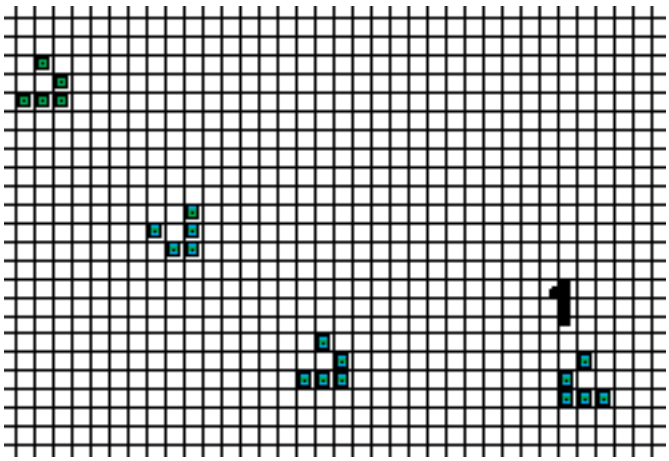

Figura 6 - Una porta logica NOT si ottiene dalla collisione di due "fasce" di alianti.

fascio che si ottiene come risultato della collisione è la codifica della sequenza 0-1-0). In modo analogo è possibile realizzare le porte logiche binarie necessarie alla costituzione di una rete combinatoria (per esempio è banale realizzare una porta XOR); più complessa risulta essere la realizzazione di elementi di memoria, ma si è dimostrato possibile ottenere anche le funzionalità di reti logiche sequenziali: tutti questi elementi sono infine sufficienti per l'implementazione di un vero calcolatore virtuale. L'estensione infinita dello spazio del gioco permette di avere una memoria potenzialmente illimitata e di conseguenza è possibile simulare una macchina di Turing universale. Avere la stessa potenza computazionale del calcolatore universale ha però una conseguenza che può forse essere considerata spiacevole: non è possibile prevedere a priori l'evoluzione di una arbitraria configurazione iniziale. Un problema di questo tipo risulta infatti essere algoritmicamente indecidibile[2]: l'unico modo di affrontarlo consiste nell'applicare le regole del gioco fino a raggiungere la situazione ricercata o una situazione incompatibile con essa, ma non si può garantire che questo modo di procedere abbia termine in un tempo finito (si noti che è un problema di finitezza e non semplicemente di durata ragionevole del tempo di computazione). Tutti i sistemi infiniti e aventi un grado di complessità non banale sono in questo senso computazionalmente irriducibili: l'unico modo di trattarli è "giocare" le loro stesse regole! Per evitarlo è necessario limitare (forse drasticamente) l'arbitrarietà delle configurazioni che costituiscono i possibili *inputs* di un programma di previsione dell'evoluzione.

Un programma che simuli l'evoluzione di un automa cellulare - indipendentemente dal fatto che sia visualizzata o meno la successione degli stati delle singole cellule - è fondamentalmente composto da un ciclo più esterno che implementa il fluire discreto del tempo e da un ciclo più interno che effettua la scansione di tutte le singole cellule dell'automa applicando a ciascuna la regola di transizione dello stato. Passando da un convenzionale calcolatore sequenziale ad una architettura "cellulare" parallela con una topologia affine a quella dell'automa da simulare (nel gioco *Life*, come in molti altri AC, la disposizione delle cellule che partecipano alla regola di transizione costituisce un intorno elementare) è possibile evitare il ciclo interno calcolando contemporaneamente per tutte le cellule lo stato successivo: la ripetizione spaziale sostituisce in questo modo una ripetizione temporale. L'elevato grado di parallelismo insito nella soluzione proposta (il tempo di calcolo dell'intera generazione successiva di organismi corrisponde al tempo impiegato per calcolare il nuovo stato della singola cellula!) è conseguenza della limitata dipendenza temporale e della estremamente localizzata inter-dipendenza spaziale dei valori da elaborare. Forse qualcuno dei lettori è un "elettronico" in grado di progettare una scheda acceleratrice per PC programmabile per la simulazione di automi cellulari. Lo stesso concetto è alla base di una eventuale implementazione del gioco con un foglio elettronico: non è difficile trovare una formula (con riferimenti relativi alle caselle che costituiscono l'intorno) che una volta "copiata" su un'intera area del tabellone generi l'evoluzione della configurazione iniziale secondo le "regole del gioco". Anche in questo modo si passa da una descrizione procedurale ad una descrizione dichiarativa delle regole, ma non se ne otterrà un incremento di velocità: i valori delle singole caselle del foglio elettronico sono infatti calcolati sequenzialmente uno alla volta.

Negli anni passati sono state proposte alcune versioni tridimensionali del gioco *Life*; nel prossimo numero esploreremo invece una classe più semplice di automi cellulari: gli AC aventi una sola dimensione spaziale. Per ingannare il tempo di questo lungo mese di attesa consideratevi come un dio che crea la vita: osservate l'evoluzione delle vostre configurazion iniziali preferite trepidando per il loro incerto futuro!

Riferimenti bibliografici

Chi vuole approfondire con spirito giocoso può consultare la seguente raccolta di articoli tratti dall'edizione italiana di *Scientific American* (Le Scienze):

- Virginio Sala (a cura di), "Divertirsi con il calcolatore: giochi, simulazione e grafica", Le Scienze ed., 1987.

2 Un esempio classico di problema algoritmicamente indecidibile è noto come "problema della fermata" (per una macchina di Turing o per un sistema formalmente equivalente). Riferendosi al calcolatore convenzionale il problema consiste nello scrivere un programma che, analizzando il testo di un altro programma scelto in modo arbitrario, decida in un tempo finito se l'esecuzione del programma preso in esame ha termine oppure no. Si dimostra facilmente che non è possibile scrivere un programma con queste caratteristiche anche limitandosi ad un linguaggio di programmazione che ha come unici costrutti l'assegnamento tra variabili ed i comandi sequenziale, condizionale ed iterativo (si noti che questa è una limitazione teorica della calcolabilità: non si tratta semplicemente di un problema difficile che non è stato ancora risolto).

[Stephen Wolfram, "Universality and Complexity in Cellular Automata", in: Farmer, Toffoli, Wolfram, "Cellular Automata", North Holland, 1984]

Automi Cellulari Lineari: impensabili universi in una sola dimensione

Giorgio Meini

Il gioco *Life* è, come si è detto, un esempio di una classe di sistemi dinamici discreti noti come Automi Cellulari (o, più brevemente: AC). Lo spazio di gioco di *Life* consiste in un piano di estensione potenzialmente illimitata composto da "cellule" quadrate o rettangolari: *Life* è quindi un automa cellulare avente 2 dimensioni spaziali e, nella simulazione al *computer*, il gioco è rappresentato dalla successione temporale delle configurazioni di stato generate dalla applicazione iterata delle regole. Chi si è divertito a "giocare" con *Life* ha senza dubbio sperimentato la difficoltà di prevedere l'evoluzione futura di una particolare configurazione iniziale (come già si è visto, nel caso generale la difficoltà diviene impossibilità: questa limitazione è intrinseca - è dovuta alla complessità del gioco stesso - e non è assolutamente superabile, non si tratta infatti di una difficoltà che - almeno in linea di principio - si può sperare di risolvere in futuro, ma di una impossibilità teorica). Nella speranza di raggiungere una maggiore comprensione - analitica e sintetica - delle dinamiche di evoluzione degli AC alcuni ricercatori (e in particolare Stephen Wolfram allo *Institute for Advanced Study* di Princeton) hanno rivolto la loro attenzione ai più semplici - sia per la geometria della disposizione spaziale delle cellule che per la topologia delle interconnessioni necessarie allo scambio delle informazioni di stato - tra gli automi cellulari: gli AC lineari. Un automa cellulare lineare è costituito da una sequenza monodimensionale - finita o infinita - di cellule che "comunicano" il proprio stato con cellule appartenenti al proprio "intorno": per intorno simmetrico di dimensione *r* si intende l'insieme costituito dalle *r* cellule a destra e dalle *r* cellule a sinistra di una cellula della sequenza, all'intorno appartiene sempre anche la cellula a cui si fa riferimento. Un AC lineare è caratterizzato, oltre che dalla dimensione *r* del proprio intorno, dal numero di stati distinti *k* che ogni cellula può assumere e dalla regola - rigidamente deterministica - che ogni cellula applica ad ogni istante discreto di tempo per calcolare il suo prossimo stato in funzione del suo stato attuale e di quello di tutte le cellule del suo intorno. Come si deduce svolgendo un semplice calcolo combinatorio il numero di regole possibili è $k^{k^{(2r+1)}}$; molti ricercatori hanno concentrato la loro attenzione su un particolare tipo di regole chiamate regole totalistiche: una regola totalistica stabilisce il nuovo valore di stato della cellula (un numero compreso tra 0, incluso, e *k*, escluso) in funzione della somma dei valori di stato di tutte le cellule appartenenti al proprio intorno (che comprende la cellula stessa); le regole totalistiche possibili sono $k^{(2r+1)}$. Una regola totalistica può essere facilmente rappresentata e codificata. Prendiamo

come esempio il caso di un AC lineare con r=1 *e* k=3: i possibili valori ottenuti dalla somma dei valori di stato delle cellule appartenenti ad un intorno sono 0, 1, 2, 3, 4, 5, 6 e la seguente è solo una tra tutte le possibili regole di transizione totalistiche

6 → 1
5 → 2
4 → 0
3 → 1
2 → 1
1 → 2
0 → 0

In un programma di simulazione una regola totalistica viene comunemente implementata in questa forma vettore, ma nella letteratura sull'argomento si trova generalmente codificata in forma numerica: l'esempio precedente con r=1 e k=3 diviene 1201120_3 si noti che il numero di stati k riportato come indice è indispensabile per decodificare la regola, la dimensione r dell'intorno è invece ricavabile dal numero di cifre che compone la codifica della regola una volta noto k).
La simulazione di un sistema di questo tipo consiste generalmente nella visualizzazione della configurazione dei singoli stati - codificati con colori distinti - di tutte le cellule dell'automa (per evitare "distorsioni ai margini" dovute al numero di cellule finito si considerano contigui gli estremi sinistro e destro della sequenza: la geometria dell'automa viene ad essere così quella della circonferenza). Convenzionalmente le configurazioni di stato corrispondenti ad istanti successivi sono visualizzate contemporaneamente e contiguamente: sullo schermo del *computer* si ottiene in questo modo la storia "per generazioni" dell'automa (formalmente si tratta di una rappresentazione grafica in 3 dimensioni dove alle 2 variabili indipendenti - lo spazio e il tempo - corrispondono le 2 dimensioni dello schermo e alla variabile dipendente - lo stato - corrisponde una codifica per colori). Non è difficile realizzare un programma che visualizzi l'evoluzione nel tempo di un automa cellulare lineare con regola di transizione totalistica: nel listato è riportata una semplice soluzione codificata in Turbo Pascal 6.0 per DOS.
Il cuore del programma è costituito due cicli "concentrici": il più esterno implementa il trascorrere del tempo come successione di istanti discreti e, al suo interno, un secondo ciclo effettua la scansione completa dello spazio lineare delle cellule (un vettore monodimensionale) per applicare ad ognuna di esse la regola di transizione (in realtà il calcolo della regola richiede una scansione di tutte le cellule del suo intorno e, di conseguenza, un ciclo ulteriore). L'essenzialità del programma che ho realizzato obbliga a modificare e compilare nuovamente il codice sorgente per impostare i parametri (r, k) e la regola di transizione totalistica dell'automa cellulare che si intende simulare; in particolare è necessario modificare il codice del programma per effettuare la scelta tra una configurazione iniziale completamente casuale o un *pattern* predefinito. Forse qualcuno dei lettori si cimenterà a trasformare questo prototipo di programma in una sofisticata versione per *Windows*.
In ossequio all'imperativo di Leibnez - *Calculemus* - "calcoliamo" il comportamento di alcuni AC lineari proposti da Wolfram in un articolo, ormai famoso, pubblicato nel 1985 sulla rivista Le Scienze [1].

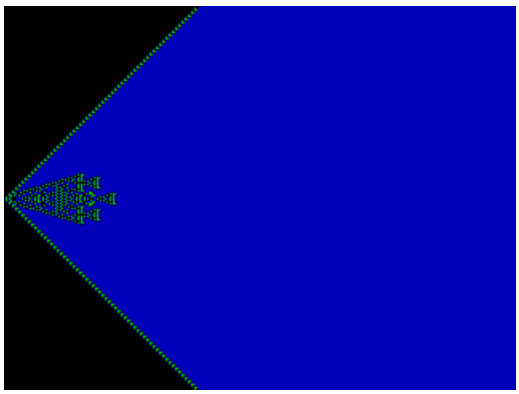

Figura 1 - Evoluzione dell'automa cellulare 12011203 (k=3, r=1, pattern: 0,0,0,...,0,1,0,0,0,...0)

La figura 1 rappresenta l'evoluzione dell'automa dell'esempio precedente: la complessa struttura simmetrica che si evolve al centro è destinata ad estinguersi, ma le due configurazioni laterali si mantengono in movimento fino alla loro collisione distruttiva, si tratta di "alianti" unidimensionali. Qualcuno dei lettori è in grado si scoprire "cannoni di alianti" in questo o in altri AC lineari? (Per aiutarsi nella ricerca si leggano le "Linee guida per il cannone ad alianti" scritte da Stephen Wolfram e riportate in appendice alla già citata raccolta di articoli curata da Virginio Sala [2].)

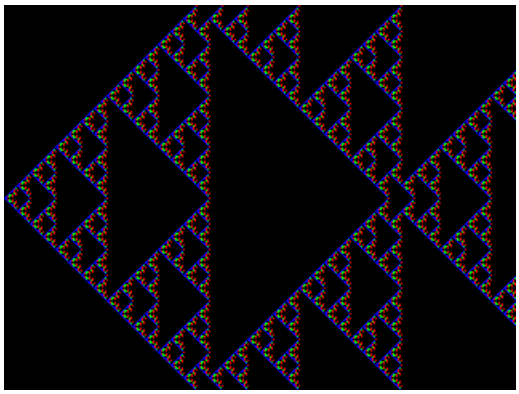

Figura 2 - Evoluzione dell'automa cellulare 000000420041 (k=5, r=1, pattern: 0,0,0,...,0,1,0,0,0,...0)

La figura 2 rappresenta invece l'evoluzione dell'automa 0000004200410_5 k=5, r=1) con *pattern* iniziale: 0,0,0,...,0,1,0,0,0,...0: è una struttura frattale dove la forma delle configurazioni si ripete per scale di osservazione (spaziali e temporali) diverse: nel caso (ipotetico) di una sequenza lineare di cellule infinita la struttura si accrescerebbe indefinitamente mantenendo la proprietà di autosomiglianza. E` interessante notare come un comportamento globale così strutturato emerga da regole di interazione esclusivamente locali: è questo uno dei segreti della vita biologica? Che sia così è la certezza di molti ricercatori di *Artificial Life* e Wolfram stesso non disdegna l'immagine di una "Natura che calcola": alle proprie conferenze ha spesso mostrato alcune conchiglie per evidenziare la somiglianza della loro conformazione con i *pattern* generati dagli AC. Il processo di auto-organizzazione di una struttura tipica della specie nel corso dello sviluppo individuale avviene mediante determinazioni locali e questo rappresenta uno dei più grandi e interessanti rompicapo della biologia moderna (Humberto Maturana e Francisco Varela), ma molti ricercatori ritengono comunque inadeguato un modello "digitale" dello spazio e del tempo per un mondo che, almeno a livello macroscopico, è indubbiamente "analogico".

Figura 3 - Evoluzione dell'automa cellulare 00000011004005 (k=5, r=1, pattern: causale)

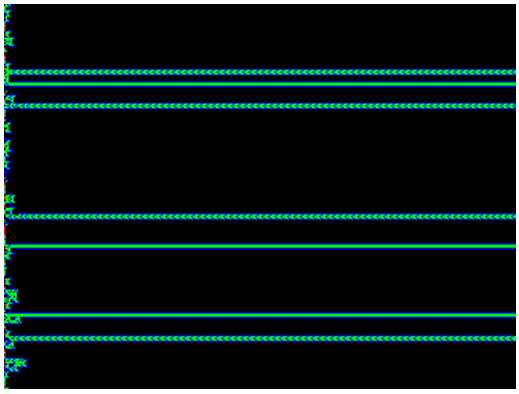

Figura 4 - Evoluzione dell'automa cellulare 00000022310005 (k=5, r=1, pattern:causale)

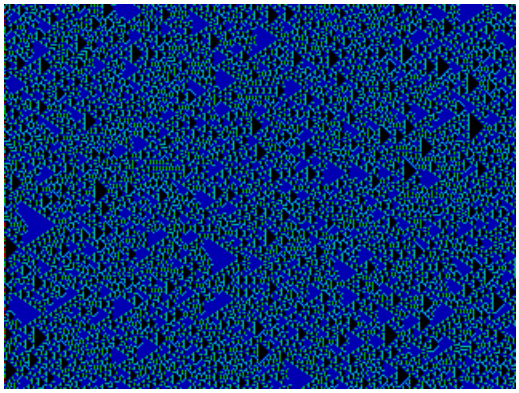

Figura 5 - Evoluzione dell'automa cellulare 00000001312105 (k=5, r=1, pattern: causale)

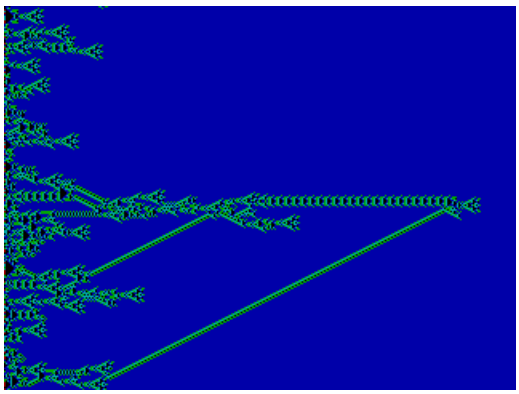

Figura 6 - Evoluzione dell'automa cellulare 00000032113105 (k=5, r=1, pattern: causale)

Le figure 3, 4 ,5 e 6 rappresentano rispettivamente l'evoluzione dei seguenti automi cellulari:

- 0000001100400_5 (k=5, r=1, *pattern*: casuale),
- 0000002231000_5 (k=5, r=1, *pattern*: casuale),
- 0000000131210_5 (k=5, r=1, *pattern*: casuale),
- 0000003211310_5 (k=5, r=1, *pattern*: casuale).

Queste 4 configurazioni sono state generate a partire da disposizioni iniziali casuali e, secondo Stephen Wolfram, rappresentano nell'ordine altrettante classi di AC aventi proprietà caratteristiche indipendenti dalle condizioni iniziali. Alla classe I appartengono automi in cui qualsiasi configurazione è destinata all'estinzione, alla classe II automi in cui si evolvono solo strutture statiche o temporalmente periodiche, alla classe III automi che realizzano *pattern* con dinamica caotica e alla classe IV - la più interessante - automi che generano strutture interagenti aventi elevata complessità locale. L'immagine riportata come intestazione rappresenta una evoluzione degli automi 000100_2, 111000_2, 101010_2 e 010100_2.

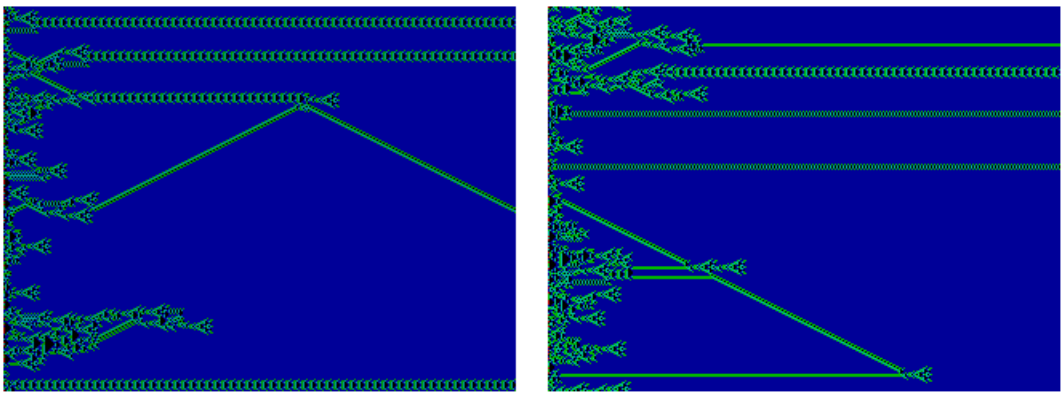

Figura 7 - Evoluzioni dell'automa cellulare 00000032113105 (k=5, r=1) a partire da distinte configurazioni iniziali

Le immagini della figura 7 rappresentano invece l'evoluzione dello stesso automa di IV classe (0000003211310_5) a partire da configurazioni iniziali casuali diverse: è possibile riconoscere alcuni "alianti" e la loro interazione con strutture periodiche.
Considerando gli AC lineari come particolari sistemi dinamici (Bruno Codenotti e Luciano Margara dell'Istituto di Matematica Computazionale del CNR di Pisa sono attualmente impegnati in ricerche che vanno in questa direzione, si veda in proposito il loro articolo riportato nella bibliografia [3]) si

ha che gli AC di classe I hanno come attrattore un punto limite, gli AC di classe II un ciclo limite, gli AC di classe III un "attrattore strano" che ne caratterizza la dinamica caotica. Come conseguenza si ha che gli AC di classe I sono praticamente insensibili a variazioni isolate delle condizioni iniziali, gli AC di calsse II hanno una sensibilità localizzata nelle immediate vicinanze della variazione e gli AC di classe III hanno una dinamica che può essere globalmente influenzata da una varizione iniziale localizzata (è questa infatti una proprietà caratteristica dei sistemi caotici): provare per credere (si provi in particolare l'automa 3311100320_4 con i seguenti semi: 000...01000...0, 000...011000...0, 000...023000...0).
E cosa dire degli automi cellulari della IV classe? La sensibilità della dinamica globale rispetto a variazioni localizzate delle condizioni iniziali è ancora maggiore che non negli AC di III classe (è possibile rendersene conto variando la configurazione iniziale dell'automa 0000003211310_5). Stephen Wolfram ritiene che gli automi appartenenti alla IV classe siano computazionalmente universali: la congettura di Wolfram è che per questa classe di AC esistano configurazioni iniziali che li rendano formalmente equivalenti ad un calcolatore convenzionale. Come abbiamo visto il mese scorso la conseguenza della universalità computazionale è la irriducibilità computazionale: non è in generale possibile prevedere a priori l'evoluzione futura di una configurazione iniziale. In questo caso l'unica previsione possibile è l'osservazione: per sapere se la successione di strutture spaziali generate dall'automa 3311100320_4 a partire dalla configurazione 000...011000...0 si estinguerà o meno non è possibile fare altro che simulare esplicitamente l'evoluzione dell'automa ed osservarne il comportamento (fino ad osservare una estinzione effettiva o una successione di configurazioni periodica nel tempo), ma così facendo non è possibile garantire una risposta in un tempo finito (la finitezza della procedura di verifica è infatti uno dei requisiti fondamentali per la decidibilità algoritmica). Non sono a conoscenza dei risultati più recenti, ma credo che la congettura avanzata da Wolfram della effettiva irriducibilità computazionale di alcuni automi cellulari lineari con regola totalistica non sia ancora stata né dimostrata né confutata (qualcuno tra i lettori può forse aggiornarci sullo stato attuale della ricerca). Esiste comunque un automa cellulare lineare con k=18 e un insieme di regole *ad hoc* di cui è stata dimostrata l'equivalenza formale con la macchina di Turing universale ed è estremamente probabile che anche AC molto più semplici siano computazionalmente irriducibili. In conclusione è questo l'insegnamento che si ricava dallo studio degli AC: un sistema relativamente complesso e rigidamente deterministico di cui si conosce senza errore la legge che ne regola il comportamento e lo stato iniziale può non essere prevedibile nella sua evoluzione fino al punto da limitare all'osservazione i possibili strumenti di analisi.
Nel passato si è ricorsi agli AC per modellare fenomeni irreversibili associati ai processi dissipativi che comunemente si incontrano in biologia e in chimica [3]. Gli AC hanno suscitato molto entusiasmo nell'ambiente della *Artificial Life*, soprattutto a causa della caratteristica relazione tra la semplicità delle componenti e la complessità del comportamento globale emergente da interazioni esclusivamente locali: si ritiene infatti che questa sinergia tra le parti ed il tutto sia uno dei tratti caratteristici della "logica del vivente". In ogni caso non tutti i sistemi sono facilmente modellizzabili con un AC: non sempre è infatti possibile individuare una decomposizione del sistema in unità elementari relativamente indipendenti l'una dall'altra ed attive contemporaneamente (in particolare non è possibile applicare questa metodologia riduzionistica a sistemi complessi - generalmente caotici - in cui lo scambio di informazioni necessario a "calcolare" la dinamica di evoluzione non sia limitato ad intorni localizzati). Un AC con regola totalistica ha, se si escludono alcuni casi particolari, una evoluzione irreversibile: dato lo stato attuale delle cellule che costituiscono l'automa non è possibile determinarne lo stato precedente. Il ricorso a regole non totalistiche permette di realizzare AC invertibili che oggi hanno assunto una grande importanza nelle applicazioni della fisica computazionale come modelli espliciti di fenomeni reversibili [3]. La proprietà di invertibilità implica un mantenimento dell'informazione iniziale nel corso dell'evoluzione dell'automa e non è quindi possibile che la relativa dinamica contempli un attrattore (l'esistenza di un attrattore implica infatti che configurazioni iniziali diverse si evolvano con "storie" che confluiscono a creare l'attrattore stesso, perdendo così la propria informazione

originale). Nonostante questa apparentemente drastica riduzione di complessità si ipotizza che anche un AC invertibile possa costituire un modello universale di calcolo: anche in questo si tratta di un "universo impensabile"!

Riferimenti bibliografici

[1] S. Wolfram, "Software nella scienza e nella matematica", Le scienze (ed. it. di *Scientific American*), Novembre 1985
[2] V. Sala (a cura di), "Divertirsi con il calcolatore: giochi simulazione e grafica", Le Scienze ed., 1987
[3] B. Codenotti e L. Margara, "Caos, complessità computazionale e parallelismo"
in: M. Capovani e B. Codenotti (a cura di), "Matematica Computazionale", Le Scienze quaderni, n. 84, Giugno 1995

Listato

```
Program LinearCA;

{
 Simulazione di automi cellulari lineari con condizioni al contorno
 periodiche e regole totalistiche.

 N: numero cellule,
 K: numero stati,
 R: dimensione intorno simmetrico.
}

Uses Crt,Graph;

Const
 Xdim=640; Ydim=480; { dimensioni schermo grafico }

 N=(Ydim div 2); K=4; R=1;
 Last=(Xdim div 2)-1;

Type
 STATE = 0..(K-1);
 TIME = 0..Last;

Const
 Next: ARRAY [0..(K-1)*(R*2+1)] OF STATE = (0,2,3,0,0,1,1,1,3,3);
  { regola totalistica }

 Pattern: ARRAY [0..N-1] OF STATE = { pattern iniziale }
  (0,0,0,0,0,0,0,0,0,0,
   0,0,0,0,0,0,0,0,0,0,
   0,0,0,0,0,0,0,0,0,0,
   0,0,0,0,0,0,0,0,0,0,
   0,0,0,0,0,0,0,0,0,0,
   0,0,0,0,0,0,0,0,0,0,
   0,0,0,0,0,0,0,0,0,0,
   0,0,0,0,0,0,0,0,0,0,
```

```
   0,0,0,0,0,0,0,0,0,0,
   0,0,0,0,0,0,0,0,0,0,
   0,0,0,0,0,0,0,0,0,0,
   0,0,0,0,0,0,0,0,0,1,
   1,0,0,0,0,0,0,0,0,0,
   0,0,0,0,0,0,0,0,0,0,
   0,0,0,0,0,0,0,0,0,0,
   0,0,0,0,0,0,0,0,0,0,
   0,0,0,0,0,0,0,0,0,0,
   0,0,0,0,0,0,0,0,0,0,
   0,0,0,0,0,0,0,0,0,0,
   0,0,0,0,0,0,0,0,0,0,
   0,0,0,0,0,0,0,0,0,0,
   0,0,0,0,0,0,0,0,0,0,
   0,0,0,0,0,0,0,0,0,0);

Var
 Gd,Gm:integer; color:word;

 OldLine,NewLine: array [0..N-1] of STATE;
 i: TIME;
 cell: 0..N-1;
 index: 0..N-1;
 CellCounter: 1..(R*2+1);
 sum: 0..(K-1)*(R*2+1);

Begin

 Randomize;
 DetectGraph(Gd,Gm);
 InitGraph(Gd,Gm,'c:\tp\bgi');
 if GraphResult<>grOk then Halt;

 for cell:=0 to N-1 do NewLine[cell]:=Pattern[cell];
 { for cell:=0 to N-1 do NewLine[cell]:=Random(K); }

 for i:=0 to Last do
  begin

   for cell:=0 to N-1 do
    begin
     color:=NewLine[cell];
     putpixel(i*2,cell*2,color);
     putpixel(i*2,cell*2+1,color);
     putpixel(i*2+1,cell*2,color);
     putpixel(i*2+1,cell*2+1,color)
    end;

   for cell:=0 to N-1 do OldLine[cell]:=NewLine[cell];
   for cell:=0 to N-1 do
    begin
     sum:=0;
     if ((cell-R)<0) then index:=N+(cell-R)
                     else index:=cell-R;
     for CellCounter:=1 to (R*2+1) do
```

```
      begin
       sum:=sum+OldLine[index];
       index:=(index+1) mod N
      end;
     NewLine[cell]:=Next[sum]
    end;
  end;

  repeat until keypressed;
  CloseGraph
end.
```