Università degli Studi della Calabria

Dottorato di Ricerca in Psicologia della Programmazione
e Intelligenza Artificiale
XV Ciclo

Tesi di Dottorato

Automi Cellulari nella modellizzazione di fenomeni complessi macroscopici e loro ottimizzazione con Algoritmi Genetici

Applicazioni in ambiente di Calcolo Parallelo
alla simulazione di colate di detrito

Il Candidato
Dott. Donato D'Ambrosio

Il Tutor
Prof. Salvatore Di Gregorio

La Coordinatrice
Prof.ssa Eleonora Bilotta

A.A. 2002-2003

A mia moglie Maria Giovanna
e a mio figlio Paolo

Indice

Capitolo 1

Introduzione

Negli corso degli ultimi decenni lo sviluppo dell'Informatica e dell'Intelligenza Artificiale ha prodotto strumenti sempre più potenti (supercalcolatori) e nuove metodologie (modelli di calcolo) per lo studio dei sistemi complessi. Benchè non esista una definizione universalmente accettata di sistema complesso e sebbene si trovino in letteratura diverse proposte, alcune sue caratteristiche distintive sono la presenza di numerosi elementi interagenti, la comparsa a livello globale di proprietà emergenti e la capacità d'auto-organizzazione [165]. Alcuni fenomeni naturali macroscopici rientrano a pieno titolo nella classe di tali sistemi; tra essi, particolarmente significativi sono i flussi detritici [59]. Si tratta di fenomeni che, in alcuni casi, possono produrre effetti altamente distruttivi e per i quali ancora oggi non esiste una caratterizzazione precisa.

L'evento del dissesto idrogeologico del maggio 1998 in Campania, che ha interessato in particolar modo la cittadina di Sarno, con le sue 161 vittime e gli ingenti danni materiali, ha riportato bruscamente all'attenzione le problematiche di mitigazione dei rischi naturali e, in particolare, di quello connesso alle colate rapide detritico-fangose. Numerosi studi di rilievo internazionale hanno tentato un'approfondita caratterizzazione delle colate detritiche, finalizzata anche alla predisposizione di misure di difesa e alla mitigazione del rischio. Seguendo approcci anche piuttosto differenti, numerose ricerche hanno visto il tentativo di comprendere le cause e riprodurre la dinamica dei flussi detritici. In alcuni casi, gli studi sono stati finalizzati alla valutazione delle soglie d'innesco meteorico e quindi della pericolosità "temporale" [26, 183, 160]; altre analisi hanno invece tentato una stima dei settori territoriali "suscettibili", ovvero della pericolosità "spaziale". In questo caso il fine della ricerca è generalmente realizzare buoni modelli di simulazione che riproducano il più fedelmente possibile la dinamica e gli effetti dei flussi detritici.

Per quanto concerne la modellistica delle colate rapide di detrito, e in generale di altri fenomeni naturali (ad esempio le colate laviche), si sono sviluppate e consolidate essenzialmente due metodologie d'indagine. La prima si basa sull'analisi delle equazioni fisiche che descrivono il fenomeno e sulla loro risoluzione attraverso metodi numerici approssimati (si veda ad esempio Laigle e Marchi [106], Liu e Lai [116], Klenov [103], Iverson [92], Iverson et al. [94], Iverson e Denlinger [93] e Toro [177]). In alternativa ai metodi analitico-deduttivi basati sull'analisi delle equazioni fisiche che descrivono il comportamento dei flussi detritici, e sulla successiva fase di risoluzione tramite metodi numerici, si è sviluppato un approccio di tipo empirico basato sul paradigma computazionale degli Automi Cellulari.

L'Automa Cellulare, proposto per la prima volta dal matematico d'origine ungherese John von Neumann [179] in uno studio che si proponeva d'indagare i meccanismi che regolano l'autoriproduzione degli organismi viventi, è un modello di calcolo parallelo la cui evoluzione è regolata da leggi puramente locali. Nell'approccio tipico che vede gli Automi Cellulari come modelli per la simulazione di fenomeni complessi, la dinamica globale dei sistemi d'interesse "emerge" dalle "interazioni locali"delle loro "componenti elementari". L'approccio è, infatti, specificamente di tipo empirico-induttivo: si parte dall'osservazione del fenomeno che si intende modellare, cercando di individuare le componenti essenziali e avanzando delle ipotesi sulla possibilità di descriverne il comportamento complessivo sulla base di leggi che determinino i meccanismi d'interazione tra le sue componenti base; in una fase successiva si definisce un modello che traduca in termini formali le leggi individuate, per poi passare a una fase di verifica, necessaria a valutare l'attendibilità del modello in relazione al fenomeno reale.

Come modelli di sistemi fisici, gli Automi Cellulari sono stati applicati, con notevoli risultati, allo studio della turbolenza nei fluidi, settore di ricerca che si è imposto all'attenzione della comunità scientifica internazionale grazie ai Gas Reticolari [80, 69, 68] e ai modelli di Boltzmann su reticolo [125, 83]. Le applicazione di questi metodi includono, in generale, fenomeni che evolvono su scala prettamente microscopica poichè identificano nelle particelle del fluido (nel caso dei Gas Reticolari), o al più nella loro densità in regioni comunque estremamente limitate (nel caso dei modelli di Boltzmann su reticolo), le componenti elementari del sistema.

Contemporaneamente alla nascita di questi modelli, il gruppo di ricerca EMPEDOCLES dell'Università della Calabria, in particolar modo nella persona del Professor Salvatore Di Gregorio, ha proposto un metodo empirico alternativo per la modellizzazione e la simulazione di fenomeni macroscopici complessi, anch'esso basato sugli Automi Cellulari, e applicato inizialmente alla simulazione delle co-

late laviche [37, 39]. Il metodo, in seguito formalizzato da Di Gregorio e Serra [59], è stato applicato successivamente alla modellizzazione di numerosi fenomeni naturali macroscopici quali le colate di detrito [10, 58, 49], i fenomeni d'erosione [47] e i processi di biorisanamento del suolo [60, 178].

A differenza dei metodi basati sull'applicazione dei Gas Reticolari, in cui la specificazione delle leggi locali che definiscono le interazioni tra le particelle del fluido non permettono l'esplicitazione dei parametri fisici del sistema (per esempio la viscosità del fluido è un parametro implicito del modello), o dei modelli di Boltzmann su reticolo, in cui generalmente i parametri del modello possono essere messi in relazione con parametri fisici (per esempio esiste una relazione ben precisa tra il tempo di rilassamento e la viscosità), il metodo empirico proposto da Di Gregorio e Serra permette, e in alcuni casi richiede, l'esplicitazione di alcuni parametri che, tuttavia, non necessariamente corrispondono a parametri fisici o possono essere messi in relazione con essi. Ad esempio, nella modellizzazione delle colate laviche e detritiche, gli effetti della viscosità sono considerati tramite uno o più parametri per i quali, però, non è definita una relazione diretta con alcun parametro fisico. L'approccio empirico del metodo, del resto, non richiede necessariamente una tale esplicitazione, prezzo che si paga, però, nella fase di verifica dell'attendibilità del modello che richiede, generalmente, numerose prove per determinare valori dei parametri "accettabili", utili per ottenere buone simulazioni in relazione al confronto con uno o, quando possibile, più eventi reali.

La ricerca discussa in questo lavoro s'inquadra nel contesto metodologico descritto da Di Gregorio e Serra con l'obbiettivo di derivare buoni modelli di simulazione per fenomeni di colate detritiche rapide, in particolare per l'applicazione nell'area di Sarno e in aree caratterizzate da proprietà geomorfologiche equivalenti. I modelli presentati sono derivati da un originario semplice modello, progressivamente arricchito sino a considerare, per la prima volta, gli effetti inerziali che caratterizzano le colate detritiche di Sarno. Il conseguente aumento di complessità ha, tuttavia, reso ancor più difficile la fase consistente nella determinazione di valori "accetabili" per i parametri dei modelli di simulazione, richiedendo l'adozione di un metodo d'ottimizzazione automatico.

Nell'ambito della modellizzazione con Automi Cellulari di fenomeni macroscopici complessi non esiste, tuttavia, alcun metodo d'ottimizzazione consolidato e, proprio per tale motivo la scelta è ricaduta sugli Algoritmi Genetici [87], algoritmi di ricerca "general purpose" che si ispirano ai meccanismi della selezione naturale e della riproduzione sessuale, già applicati con buoni risultati in differenti contesti scientifici e ingegneristici. L'Algoritmo Genetico è un algoritmo iterativo che deter-

mina l'evoluzione di una "popolazione" di soluzioni candidate di un dato problema di ricerca (nel caso specifico la ricerca di un buon insieme di parametri dei modelli di simulazione). A ogni iterazione, ogni soluzione candidata è valutata per stabilire i "rapporti di forza" tra gli "individui" della popolazione in modo che i più abili siano scelti preferenzialmente per generare nuove soluzioni candidate. La valutazione di un individuo necessita di un'intera simulazione del modello che si intende ottimizzare, adottando i parametri specificati dalla particolare soluzione candidata che si sta valutando. Questo, tuttavia, può richiedere tempi di calcolo eccessivi, rendendo l'esecuzione dell'Algoritmo Genetico estremamente lenta; i primi esperimenti d'ottimizzazione eseguiti in ambiente di calcolo sequenziale hanno richiesto, infatti, alcuni mesi per restituire il risultato dell'elaborazione. L'adozione del Calcolo Parallelo ha permesso, tuttavia, d'abbattere considerevolmente i tempi d'esecuzione consentendo, inoltre, un primo studio qualitativo sulla dinamica dell'Algoritmo Genetico nella ricerca della soluzione ottimale, utile per comprendere l'efficacia del metodo d'ottimizzazione.

La tesi è organizzata nel seguente modo. Gli Algoritmi Genetici e gli Automi Cellulari sono presentati, rispettivamente, nel secondo e nel terzo capitolo, insieme ai risultati teorici più importanti e ad alcune applicazioni ritenute particolarmente significative. Il quarto capitolo discute l'applicazione degli Automi Cellulari alla modellizzazione di fenomeni macroscopici complessi, descrivendo il metodo empirico di Di Gregorio e Serra e presentando un esempio d'applicazione alla simulazione dei flussi lavici etnei. Nello stesso capitolo è successivamente presentata nel dettaglio l'applicazione del metodo alla simulazione dei flussi detritici, ricerca a partire dalla quale il candidato ha contribuito significativamente. Nel quinto capitolo sono presentati due nuovi modelli per la simulazione di flussi detritici rapidi caratterizzati da forti effetti inerziali, insieme alla loro ottimizzazione con Algoritmi Genetici. Nello stesso capitolo sono descritti i risultati di due serie di esperimenti d'ottimizzazione relativi al caso di studio di Curti (Sarno), eseguiti il primo in ambiente di calcolo sequenziale, il secondo in ambiente di calcolo parallelo. Ancora nel quinto capitolo sono presentati i risultati di uno studio qualitativo sulla dinamica dell'Algoritmo Genetico, pensato per valutare l'efficacia del metodo nell'ottimizzazione dei modelli di simulazione considerati. Il sesto capitolo conclude, infine, la tesi con una discussione generale sul lavoro svolto.

Capitolo 2

Algoritmi Genetici

2.1 Introduzione

Gli Algoritmi Genetici (AG) [87] sono algoritmi di ricerca e modelli scientifici dell'evoluzione che si ispirano ai meccanismi della selezione naturale e della riproduzione sessuale. L'idea alla base degli AG è simulare l'evoluzione di una popolazione di individui, che rappresentano soluzioni candidate di uno specifico problema, favorendo la sopravvivenza e la riproduzione dei migliori.

Gli individui sono solitamente considerati a due differenti livelli, genotipico e fenotipico. Al livello genotipico gli individui, detti genotipi o cromosomi, sono rappresentati tramite una particolare struttura dati, solitamente un array o un albero, opportunamente scelta per codificare nel modo più conveniente possibile le soluzioni candidate. Gli elementi costituenti la struttura dati sono detti geni, ognuno dei quali può assumere un certo numero di valori, detti alleli. Al livello fenotipico gli individui, detti fenotipi, sono invece l'espressione, cioè la decodifica, delle proprie rappresentazioni genotipiche nelle corrispondenti soluzioni candidate.

Nella metafora evolutiva, il problema da risolvere rappresenta l'ambiente cui gli individui devono adattarsi per sopravvivere e riprodursi e, in tale contesto, un individuo è considerato migliore di un altro o, equivalentemente, più adatto, se la soluzione che codifica risolve meglio il problema. I membri della popolazione iniziale $P(t = 0)$, solitamente generati in modo casuale, sono valutati tramite una funzione, detta di fitness, che ne misura il valore di adattività, o di fitness, e i migliori sono selezionati per la riproduzione e copiati nel, così detto, mating pool. Gli individui della nuova popolazione $P(t + 1)$ sono generati dagli individui del mating pool attraverso semplici operatori random, detti operatori genetici,

```
AG{
      t=0;
      inizializza la popolazione P(0)
      valuta la popolazione P(0)
      mentre(non(criterio di fermata)){
            t=t+1
            crea il mating pool MP da P(t-1)
            crea la popolazione P(t) da MP
            valuta la popolazione P(t)
      }
}
```

Figura 2.1: Schema iterativo base di un algoritmo genetico.

ispirati alla riproduzione sessuale e alla mutazione. Il ciclo di valutazione, selezione, riproduzione sessuale e mutazione è iterato per un certo numero di generazioni, fino al raggiungimento di un dato criterio di fermata, come il raggiungimento del numero massimo di iterazioni consentite oppure la convergenza a una soluzione il cui errore rispetto alla soluzione ottimale risulti minore di una certa soglia prestabilita. La figura 2.1 illustra lo schema iterativo base dell'algoritmo genetico.

Prima di analizzare nei dettagli l'originario modello di Holland e i risultati teorici più importanti su esso derivati, il paragrafo 2.2 descrive brevemente la storia degli AG dalla loro nascita a oggi.

2.2 Breve storia degli Algoritmi Genetici

Gli AG nacquero in un contesto in cui un numero sempre crescente di ricercatori e studiosi aveva iniziato a interessarsi dei sistemi naturali come fonte d'ispirazione per la realizzazione di algoritmi di ottimizzazione per problemi ingegneristici.

Nei primi anni '60 John Holland, con l'obbiettivo di realizzare sistemi capaci di auto-adattamento, si interessò dei principi che regolano l'evoluzione dei sistemi adattivi naturali ipotizzando che la competizione e l'innovazione fossero i meccanismi fondamentali tramite cui gli individui acquisiscono le capacità di adattarsi ad ambienti che cambiano nel tempo e rispondere a eventi inattesi [86].

Dalla metà degli anni '60 vennero realizzati semplici sistemi computazionali che, tuttavia, già presentavano caratteristiche comuni agli AG così come li conosciamo oggi: una popolazione di individui era fatta evolvere nel tempo e i meccanismi di derivazione erano semplici astrazioni di operatori genetici.

Nonostante le iniziali perplessità della comunità d'Intelligenza Artificiale, Hol-

land continuò il suo lavoro sui sistemi adattivi introducendo la nozione di schema, di parallelismo implicito e dimostrando il Teorema Fondamentale degli Algoritmi Genetici [87]. Così, diversamente da quanto accadde per le altre due principali tecniche computazionali ispirate all'evoluzione che naquero all'incirca nello stesso periodo, le Strategie Evolutive [154] e per la Programmazione Evolutiva [65], Holland pose delle significative basi teoriche per il suo modello su cui si sono basati moltissimi dei successivi studi sugli AG.

Successivamente De Jong [54] si occupò dell'analisi e dell'applicazione di alcuni semplici modelli di algoritmi genetici per l'ottimizzazione di un gruppo di funzioni mettendo in risalto il fatto che tali semplici modelli potessero essere utilizzati in maniera significativa anche come algoritmi d'ottimizzazione.

L'interesse intorno agli AG cominciò lentamente a crescere fino al 1989, anno in cui David Goldberg pubblicò il libro *Genetic Algorithms in Search Optimisation and Machine Learning* [72]. Il testo di Goldberg, oggi considerato un classico, ebbe alla sua uscita l'effetto di catalizzare l'attenzione della comunità scientifica sugli Algoritmi Genetici poichè presentava la teoria e le applicazioni degli AG in una forma chiara, precisa e di facile comprensione [53].

Il periodo dal 1990 a oggi è stato caratterizzato da un'enorme crescita della comunità degli Algoritmi Genetici e nuove applicazioni degli AG hanno interessato un gran numero di nuove aree di ricerca.

2.3 Il modello di Holland

Il modello di algoritmo genetico proposto per la prima volta da Holland [87] intorno alla metà degli anni '60 è un algoritmo iterativo che opera su una popolazione di n stringhe di bit di lunghezza l fissata ($n, l \in \mathbb{N}$) in cui ogni stringa (genotipo) è la codifica binaria di una soluzione candidata (fenotipo) di un particolare problema di ricerca. Si osservi che l'insieme delle stringhe binarie di lunghezza l ha 2^l elementi e che tale numero, crescendo esponenzialmente al crescere di l, può essere anche molto grande. Tale insieme rappresenta lo spazio di ricerca dell'algoritmo genetico, cioè lo spazio che l'algoritmo genetico deve esplorare per risolvere il problema di ricerca.

La funzione di fitness f dell'algoritmo genetico, definita dall'utente in relazione al particolare problema di ricerca, associa a ogni genotipo g_i (i=1,...,n) un valore di adattività, o di fitness, $f_i = f(g_i)$. Per determinare tale valore la funzione di fitness decodifica il genotipo nel corrispondente fenotipo e lo testa sul problema

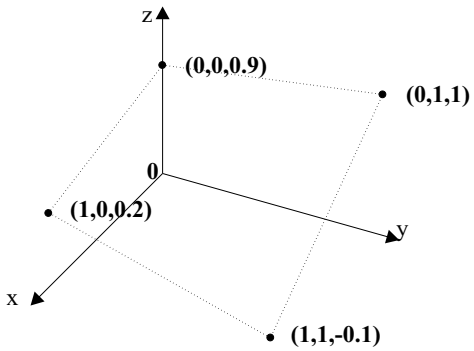


Figura 2.2: Esempio di paesaggio d'idoneità di un algoritmo genetico con genotipi binari di 2 bit. Nel caso specifico la terna (1,0,0.2) indica che il valore di fitness associato al punto (1,0) è $f(1,0) = 0.2$, essendo f la funzione di fitness dell'algoritmo genetico. Lo stesso vale per gli altri tre punti.

dato restituendo un valore, solitamente un numero reale, che ne rappresenta la capacità di risolvere il problema. Il grafico dei valori di fitness sui punti dello spazio di ricerca è detto paesaggio d'idoneità (fitness landscape). In figura 2.2 è illustrato un possibile paesaggio d'idoneità di un algoritmo genetico con genotipi binari di lunghezza $l = 2$.

L'originario modello di Holland è oggi noto come modello con schema generazionale poichè a ogni generazione rimpiazza tutti gli n individui della popolazione con altrettanti discendenti, mentre il metodo di selezione è noto come selezione proporzionale poichè seleziona gli individui da riprodurre con probabilità proporzionale alla fitness. Gli operatori genetici che Holland utilizzò sono il crossover, oggi noto come crossover a punto singolo, la mutazione e l'inversione. L'operatore d'inversione è stato, tuttavia, raramente utilizzato nelle applicazioni e ancor più di rado analizzato dal punto di vista teorico e per tale motivo, al contrario del metodo di selezione e degli altri operatori genetici, non è descritto nei paragrafi successivi.

2.3.1 Selezione

Proporzionalmente al valore di fitness f_i, ai genotipi g_i sono associate le probabilità

$$p_{selection,i} = \frac{f_i}{\sum_{j=1}^{n} f_j} \tag{2.1}$$

utilizzate per costruire una sorta di roulette di probabilità. Per esempio, se la popolazione è composta dagli $n = 4$ individui A_1, A_2, A_3 e A_4, con le rispettive probabilità di selezione $p_{selection,1} = 0.12$, $p_{selection,2} = 0.18$, $p_{selection,3} = 0.3$ e

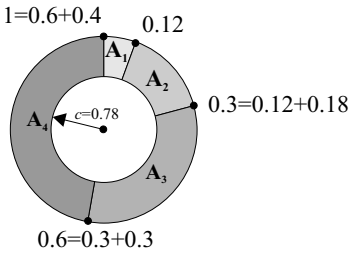


Figura 2.3: Esempio di roulette per l'operatore di selezione. I quattro individui A_1, A_2, A_3 e A_4, con probabilità di selezione 0.12, 0.18, 0.3 e 0.4, occupano uno spicchio di roulette di ampiezza pari alla propria probabilità di selezione. Nell'esempio l'operatore di selezione genera il numero casuale $c = 0.78$ e l'individuo A_4 viene scelto e inserito nel mating pool.

$p_{selection,4} = 0.4$, la corrispondente roulette avrà la forma mostrata in figura 2.3. L'operatore di selezione genera un numero casuale $c \in [0, 1]$ e seleziona l'individuo la cui porzione di roulette ne contiene il valore. Per esempio se $c = 0.78$, con riferimento alla figura 2.3, l'individuo selezionato è A_4 poichè la porzione di roulette relativa ad A_4 specifica i valori dell'intervallo $[0.6, 1]$ e il valore di c ricade in tale intervallo. Quando un individuo è selezionato ne viene creata una copia e quest'ultima viene inserita nel mating pool. Una volta che il mating pool è riempito con n copie di individui della popolazione $P(t)$, gli individui della nuova popolazione $P(t + 1)$ sono ottenuti come loro discendenti attraverso l'applicazione degli operatori genetici. L'operatore di selezione determina, dunque, quali individui della vecchia popolazione hanno la possibilità di generare dei discendenti e poichè gli individui con fitness alta sono quelli favoriti, potendo contare un numero di copie maggiore rispetto a quelli con fitness più bassa, l'operatore di selezione gioca, nel contesto dell'algoritmo genetico, il ruolo della selezione naturale.

2.3.2 Crossover

Si scelgono a caso due individui nel mating pool, detti genitori, e un punto di taglio, detto punto di crossover, su di essi. Le porzioni di genotipo alla destra del punto di crossover vengono scambiate generando due discendenti. La figura 2.4 illustra un esempio di crossover tra due genotipi binari. L'operatore di crossover è applicato, in accordo a una prefissata probabilità $p_{crossover}$, $\frac{n}{2}$ volte in modo da ottenere n discendenti; nel caso in cui il crossover non sia applicato, i discendenti coincidono con i genitori. Si osservi che, così come l'operatore di selezione gioca,

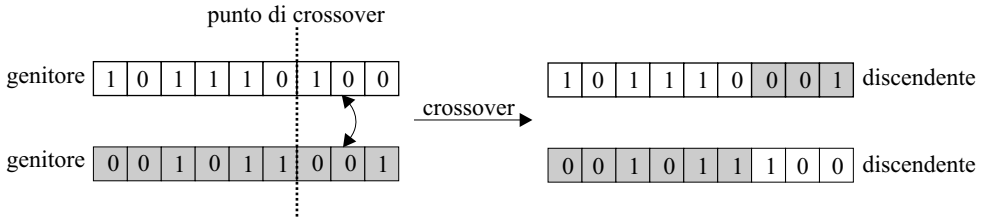


Figura 2.4: Esempio di crossover. Viene generato casualmente un punto di taglio nella posizione 6 (dopo il sesto gene) e le due stringhe genitori sono ricombinate generando due discendenti.

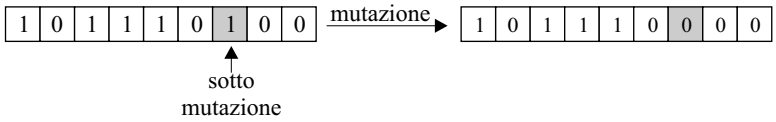


Figura 2.5: Esempio di mutazione. Il settimo gene è l'unico soggetto a mutazione e cambia il suo valore da 1 in 0. Tutti i valori degli altri geni rimangono invariati.

nel contesto dell'algoritmo genetico, il ruolo della selezione naturale, il crossover rappresenta una metafora della riproduzione sessuale in cui il materiale genetico dei discendenti risulta essere una ricombinazione di quello dei genitori.

2.3.3 Mutazione

In funzione di una prefissata e usualmente piccola probabilità $p_{mutation}$, il valore dei bit di ogni individuo viene cambiato (da 0 in 1 o viceversa). La figura 2.5 illustra un esempio di mutazione su un genotipo binario. L'operatore di mutazione rappresenta il fenomeno genetico della rara variazione di elementi del genotipo negli esseri viventi durante l'evoluzione.

2.4 Fondamenti teorici degli Algoritmi Genetici

Negli anni '70 John Holland [87] ottenne alcuni importanti risultati teorici sul suo modello su cui si sono basati la gran parte dei successivi studi teorici sugli AG. I paragrafi successivi, con l'aiuto di alcuni esempi, ne illustrano il significato e ne forniscono la dimostrazione.

2.4.1 Schemi e parallelismo implicito

Definizione 2.4.1. Sia $V = \{0, 1\}$ l'alfabeto attraverso cui sono costruiti gli individui dell'algoritmo genetico. Si definisce schema una stringa della stessa lunghezza l degli individui dell'algoritmo genetico, costruiti sull'alfabeto esteso $V+ = \{0, 1, *\}$.

Osservazione 2.4.1. Il numero di stringhe che si possono costruire su V è 2^l, mentre il numero di schemi è $(2 + 1)^l$. Per esempio se $l = 5$, si avranno $k^l = 2^5 = 32$ individui e $(k + 1)^l = 3^5 = 243$ schemi.

Uno schema rappresenta un insieme di stringhe che hanno un qualsiasi simbolo di V nelle corrispondenti posizioni in cui compare il simbolo $*$ o, equivalentemente, rappresenta un insieme di stringhe con caratteristiche comuni nelle corrispondenti posizioni in cui non compare il simbolo $*$.

Esempio 2.4.1. Lo schema $H_1 = *0000$ rappresenta l'insieme costituito dalle due stringhe aventi uno dei due simboli di V nella posizione 0 e il simbolo 0 nelle posizioni 2, 3, 4 e 5, cioè

$$H_1 = \{00000, 10000\}$$

Lo schema $H_2 = *111*$ rappresenta l'insieme costituito dalle quattro stringhe aventi uno dei due simboli di V nelle posizioni 1 e 5 e il simbolo 1 nelle posizioni 2, 3 e 4, cioè

$$H_2 = \{01110, 01111, 11110, 11111\}$$

Lo schema $H_3 = 0 * 1 * *$ rappresenta l'insieme costituito dalle otto stringhe aventi uno dei due simboli di V nelle posizioni 2, 4 e 3, uno 0 nella posizione 1 e un 1 nella posizione 3, cioè

$$H_3 = \{00100, 00101, 00110, 00111, 01100, 01101, 01110, 01111\}$$

Per comprendere meglio il significato di schema consideriamo il seguente esempio.

Esempio 2.4.2. Sia $V = \mathbb{R}^+ \cup \{0\}$, ed $l = 3$. Allora lo spazio di ricerca dell'algoritmo genetico è $S = [\mathbb{R}^+]^3 \cup \{\mathbf{0}\}$.
Lo schema $H_1 = * * 0$ rappresenta il sottoinsieme di S formato dai punti del piano $\pi = \{(x, y, z) \in S \ / \ z = 0\}$ (figura 2.6). Infatti i simboli $*$ nella prima e seconda posizione dello schema indicano che le coordinate x e y possono assumere un qualsiasi valore di $V = \mathbb{R} + \cup\{0\}$, mentre la coordinata z è fissata a 0.
Lo schema $H_2 = * * 1$ rappresenta il sottoinsieme di S formato dai punti del piano

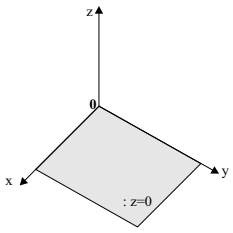


Figura 2.6: Il sottoinsieme $\pi = \{(x, y, z) \in S \ / \ z = 0\}$ dello spazio di ricerca $S = [\mathbb{R}^+]^3 \cup \{\mathbf{0}\}$ dell'algoritmo genetico definito dallo schema $H_1 = * * 0$.

$\pi = \{(x, y, z) \in S \ / \ z = 1\}$ (figura 2.7). Infatti, come prima, i simboli $*$ nella prima e seconda posizione dello schema indicano che le coordinate x e y possono assumere un qualsiasi valore di $V = \mathbb{R}^+ \cup \{0\}$, mentre la coordinata z è fissata a 1.

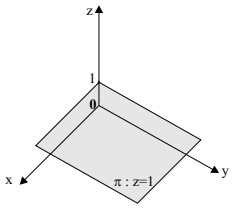


Figura 2.7: Il sottoinsieme $\pi = \{(x, y, z) \in S \ / \ z = 1\}$ dello spazio di ricerca $S = [\mathbb{R}^+]^3 \cup \{\mathbf{0}\}$ dell'algoritmo genetico definito dallo schema $H_2 = * * 1$.

Lo schema $H_3 = 1 * 1$ rappresenta il sottoinsieme di S formato dai punti della retta $r = \{(x, y, z) \in S \ / \ x = 1 \ e \ z = 1\}$ (figura 2.8). Infatti il simbolo $*$ nella posizione 2 dello schema indica che la coordinata y può assumere un qualsiasi valore di $V = \mathbb{R} + \cup\{0\}$, mentre le coordinate x e z sono fissate a 1.

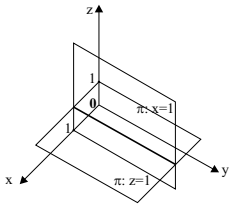


Figura 2.8: Il sottoinsieme $r = \{(x, y, z) \in S \ / \ x = 1 \ e \ z = 1\}$ dello spazio di ricerca $S = [\mathbb{R}^+]^3 \cup \{\mathbf{0}\}$ dell'algoritmo genetico definito dallo schema $H_3 = 1 * 1$.

Gli schemi rappresentano, dunque, sottoinsiemi dello spazio di ricerca dell'algoritmo genetico.

Teorema 2.4.1. *Il numero di schemi di un algoritmo genetico presenti in una popolazione di n individui di lunghezza l costruiti sull'alfabeto* $V = \{0, 1\}$ *è compreso tra* 2^l *e* $n2^l$.

Dimostrazione. Consideriamo una generica stringa $A = a_1 a_2 \dots a_l$ tale che $a_i \in V = \{0, 1\} \quad \forall i \in \{1, 2, \dots, l\}$. Essa è un membro di ogni schema che abbia nelle corrispondenti posizioni di A o lo stesso simbolo di A oppure il simbolo $*$. Per esempio A è membro dello schema $H = *a_2 * * \dots * a_l$. Dunque, ogni schema che contenga A è vincolato ad avere in ogni posizione uno tra due simboli: il corrispondente simbolo di A, oppure $*$. Il loro numero è, pertanto, 2^l. Può succedere che, per esempio quando tutte le stringhe della popolazione sono ugluali, tutti gli individui siano membri degli stessi schemi; in tal caso il numero di schemi della popolazione è 2^l. All'opposto, quando tutte le stringhe della popolazione sono membri di schemi diversi, il loro numero è $n2^l$. □

Il teorema precedente afferma che l'algoritmo genetico, pur elaborando esplicitamente solo n individui, processa implicitamente un numero molto maggiore di schemi. Questo significa che è garantita la presenza di qualche soluzione candidata in un numero compreso tra 2^l e $n2^l$ sottoinsiemi differenti dello spazio di ricerca, garantendo un'esplorazione abbastanza efficace dello spazio delle soluzioni del problema. Tale caratteristica è chiamata parallelismo implicito.

2.4.2 Il Teorema Fondamentale degli Algoritmi Genetici

Definizione 2.4.2. Si definisce ordine di uno schema H, e si indica con $o(H)$, il numero di posizioni fisse presenti nel modello.

Esempio 2.4.3.

$$o(011 * 1 * *) = 4$$

$$o(0 * * * * * *) = 1$$

Definizione 2.4.3. Si definisce lunghezza caratteristica di uno schema H, e si indica con $\delta(H)$, la distanza tra la prima e l'ultima posizione fissa dello schema.

Esempio 2.4.4.

$$\delta(h_1 h_2 h_3 h_4 h_5 h_6 h_7) = \delta(011 * 1 * *) = 5 - 1 = 4$$

$$\delta(h_1 h_2 h_3 h_4 h_5 h_6 h_7) = \delta(0 * * * * * *) = 1 - 1 = 0$$

Consideriamo ora gli effetti della selezione, del crossover e della mutazione sul numero di rappresentanti degli schemi dell'algoritmo genetico. Supponiamo che a una data generazione t esistano $m = m(H,t)$ rappresentanti di un particolare schema H nella popolazione P(t). Come descritto nel paragrafo 2.3, formula 2.1, ogni individuo, quindi anche ogni rappresentante di H, è selezionato per la riproduzione con probabilità $p_{selection,i} = \frac{f_i}{\sum_{j=1}^{n} f_j}$. Dunque, la probabilità che una stringa dello schema venga selezionata a ogni tentativo è

$$\sum_{i=1}^{m} p_{selection,i} = \frac{\sum_{i=1}^{m} f_i}{\sum_{j=1}^{n} f_j}$$

Poichè l'operatore di selezione è eseguito n volte, il numero atteso di rappresentanti di H è

$$m(H,t+1) = n\sum_{i=1}^{m} p_{selection,i} = n\frac{\sum_{i=1}^{m} f_i}{\sum_{j=1}^{n} f_j} = \frac{\sum_{i=1}^{m} f_i}{\bar{f}}$$

dove $\bar{f}$ è la fitness media della popolazione $P(t)$. Moltiplicando ambo i membri per $m(H,t)$ si ha

$$m(H,t+1) = \frac{m(H,t)}{m(H,t)}\frac{\sum_{i=1}^{m} f_i}{\bar{f}} = m(H,t)\frac{f(H)}{\bar{f}} \tag{2.2}$$

dove $f(H)$ è la fitness media degli individui rappresentanti dello schema H. La formula 2.2 indica che se la fitness media dei rappresentanti di uno schema H è maggiore della fitness media della popolazione, allora il numero di rappresentanti dello schema aumenta, altrimenti diminuisce.

Per determinare la velocità con cui varia il numero di rappresentanti scriviamo

$$f(H) = \bar{f} + c\bar{f}$$

dove c è un costante reale, minore di uno se la fitness media dello schema è minore della fitness media della popolazione, maggiore di uno altrimenti. Allora

$$m(H,t+1) = m(H,t)\frac{\bar{f} + c\bar{f}}{\bar{f}} = m(H,t)(1+c)$$

per cui si avrà

$$m(H,t) = m(H,t-1)(1+c) = m(H,t-2)(1+c)^2 = \ldots$$

$$\ldots = m(H,0)(1+c)^t \tag{2.3}$$

Dunque, il numero di rappresentanti di uno schema cresce, o decresce, esponenzialmente durante le generazioni dell'algoritmo genetico a seconda che la fitness media dello schema sia maggiore o minore della fitness media della popolazione.

Consideriamo ora un individio A e due schemi, H_1 e H_2, di cui A sia rappresentante

$$A = 0111000$$

$$H_1 = *1****0$$

$$H_2 = ***10**$$

e supponiamo che A sia sogetto a crossover con punto di taglio nella posizione 3, cioè dopo il terzo gene

$$A = 011 \mid 1000$$

$$H_1 = *1* \mid ***0$$

$$H_2 = *** \mid 10**$$

Osserviamo che i discendenti di A potrebbero non essere rappresentanti dello schema H_1, a meno che A non si incroci con un individuo che abbia anch'esso un 1 nella posizione 2 o uno 0 nella posizione 7. Infatti se A si incrocia con l'individuo $B = 1000101$, che non ha nè 1 nella posizione 2 nè 0 nella posizione 7, i due discendenti, A' e B', non saranno rappresentanti di H_1:

$$\begin{array}{l} A = 0\mathbf{1}1 \mid 100\mathbf{0} \\ B = 100 \mid 0101 \end{array} \Rightarrow \begin{array}{l} A' = 0\mathbf{1}10101 \notin H_1 \\ B' = 100100\mathbf{0} \notin H_1 \end{array}$$

mentre, se A si incrocia con l'individuo $B = 1100101$, che ha 1 nella posizione 2 e non ha 0 nella posizione 7, uno dei discendenti sarà ancora rappresentante di H_1:

$$\begin{array}{l} A = 0\mathbf{1}1 \mid 100\mathbf{0} \\ B = 1\mathbf{1}0 \mid 0101 \end{array} \Rightarrow \begin{array}{l} A' = 0\mathbf{1}10101 \notin H_1 \\ B' = 1\mathbf{1}01000 \in H_1 \end{array}$$

Lo stesso accade se A si incrocia con un individuo che non ha 1 nella posizione 2 e ha 0 nella posizione 7. Ovviamente, Se A si incrocia con un individuo anch'esso rappresentante di H_1, entrambi i discendenti saranno ancora rappresentanti di H_1.

Al contrario, poichè le posizioni fisse di H_2 cadono tutte a destra del punto di taglio, in ogni caso almeno un discendente di A sarà ancora rappresentante di H_2. Anche se A si incrocia con un individuo che non ha nè 1 nella posizione 4

nè 0 nella posizione 5, per esempio $B = 1000101$, uno dei due discendenti sarà comunque rappresentante di H_2:

$$\begin{array}{l} A = 011 \mid \mathbf{10}00 \\ B = 110 \mid 0101 \end{array} \Rightarrow \begin{array}{l} A' = 0110101 \notin H_2 \\ B' = 110\mathbf{10}00 \in H_2 \end{array}$$

Lo schema H_1 ha minori probabilità di sopravvivere al crossover rispetto ad H_2 perchè ha maggiori probabilità che il punto di taglio cada tra le due posizioni fisse estreme. Infatti $\delta(H_1) = 7 - 2 = 5$, e il punto di taglio può cadere con eguale probabilità in uno degli $l - 1 = 7 - 1 = 6$ posizioni. Allora le probabilità che H_1 e H_2 vengano distrutti, cioè che perdano rappresentanti, per effetto del crossover sono

$$p_d(H_1) \leq \frac{\delta(H_1)}{l-1} = \frac{5}{6}$$

$$p_d(H_2) \leq \frac{\delta(H_2)}{l-1} = \frac{1}{6}$$

dove il simbolo $\leq$ è giustificato dal fatto che, anche se il punto di taglio cade all'interno delle posizioni fisse, lo schema può comunque sopravvivere.

Pertanto, definiamo la probabilità che uno scema H venga distrutto come

$$p_d(H) \leq \frac{\delta(H)}{l-1}$$

Se il crossover è eseguito con probabilità $p_{crossover}$, la formula precedente diventa

$$p_d(H) \leq p_{crossover}\frac{\delta(H)}{l-1}$$

Pertanto, la probabilità che lo schema H sopravviva al crossover è

$$p_s = 1 - p_d \geq 1 - p_{crossover}\frac{\delta(H)}{l-1} \tag{2.4}$$

L'espressione precedente afferma che gli schemi che anno alte probabilità di sopravvivere al crossover sono quelle che hanno lunghezze caratteristiche piccole. Moltiplicando la 2.2 con la 2.4 otteniamo il numero atteso di rappresentanti dello schema H per effetto della selezione e del crossover:

$$m(H, t+1) >= m(H, t)\frac{f(H)}{\bar{f}}(1 - p_{crossover}\frac{\delta(H)}{l-1}) \tag{2.5}$$

Affinchè uno schema H sopravviva anche all'operatore di mutazione, tutte le $o(H)$ posizioni fisse devono rimanere immutate. Poichè la probabilità che una

singola posizione fissa rimanga immutata è $1-p_{mutation}$ e questo deve succedere per tutte le $o(H)$ posizioni fisse, la probabilità che lo schema H sopravviva all'operatore di selezione è:

$$p'_s = (1 - p_{mutation})^{o(H)}$$

Se $p_{mutation} \ll 1$, allora

$$p'_s = (1 - p_{mutation})^{o(H)} \approx 1 - o(H)p_{mutation} \tag{2.6}$$

Moltiplicando la 2.5 per la 2.6 otteniamo il numero atteso di rappresentanti dello schema H per effetto della selezione del crossover e della mutazione:

$$m(H, t+1) >= m(H, t)\frac{f(H)}{\bar{f}}(1 - p_{crossover}\frac{\delta(H)}{l-1})(1 - o(H)p_{mutation}) =$$

$$= m(H, t)\frac{f(H)}{\bar{f}}(1 - o(H)p_{mutation} - p_{crossover}\frac{\delta(H)}{l-1} +$$

$$+p_{crossover}\frac{\delta(H)}{l-1}o(H)p_{mutation})$$

Il termine $p_{crossover}\frac{\delta(H)}{l-1}o(H)p_{mutation}$ si può trascurare perchè abbastanza piccolo, per cui la formula precedente diventa

$$m(H, t+1) >= m(H, t)\frac{f(H)}{\bar{f}}(1 - p_{crossover}\frac{\delta(H)}{l-1} - o(H)p_{mutation}) \tag{2.7}$$

Il risultato precedente prende il nome di Teorema Fondamentale degli Algoritmi Genetici. Il numero atteso di rappresentanti di schemi caratterizzati da fitness media alta e lunghezza caratteristica piccola, chiamati da Goldberg blocchi costituenti [72], aumenta esponenzialmente (eq. 2.3) di generazione in generazione.

In conclusione, l'algoritmo genetico ha la capacità di esplorare un numero compreso tra 2^l e $n2^l$ schemi per ogni generazione (parallelismo implicito) e di concentrare il maggior numero di individui nelle zone dello spazio di ricerca più promettenti poichè premia gli schemi caratterizzati da fitness alta (Teorema Fondamentale degli Algoritmi Genetici).

2.5 Varianti del modello di Holland

Il modello proposto da Holland ha ispirato tutti gli studi teorici e applicativi sugli AG. E' chiaro comunque che, quando si tratta di risolvere problemi reali, l'algoritmo genetico proposto da Holland è, per molti aspetti, limitativo: non è possibile

codificare convenientemente tutti i problemi con stringhe di bit, non sempre la selezione proporzionale all'idoneità è il metodo migliore e non sempre gli operatori genetici proposti da Holland sono i più efficaci e i più appropriati [126]. Per tali ragioni, a partire dalla fine degli anni '80, sono stati proposti nuovi modelli di AG che si differenziano dall'originario modello di Holland negli schemi di codifica del genotipo, negli operatori genetici utilizzati e nelle strategie di selezione adottate.

2.5.1 Codifiche e operatori genetici

In linea di principio è sempre possibile codificare le soluzioni candidate di un problema di ricerca attraverso stringhe binarie. Tuttavia, per la risoluzione di alcuni problemi risulta più naturale utilizzare rappresentazioni di livello più alto e definire operatori di crossover e mutazione in grado di operare su tali rappresentazioni. Nelle applicazioni pratiche le codifiche più diffuse sono quella binaria, quella basata su numeri reali e la codifica ad albero.

Codifica binaria

La codifica binaria è, probabilmente, la più utilizzata nelle applicazioni pratiche, sia per motivi storici, sia perchè su essa sono stati derivati i risultati teorici più importanti.

La struttura dati utilizzata è un vettore di bit di lunghezza l, cui corrisponde uno spazio di ricerca di 2^l possibili soluzioni. L'uso della codifica binaria richiede la specificazione di una funzione che decodifichi il genotipo, o parti di esso, nel corrispondente fenotipo. Per esempio, la seguente equazione decodifica un genotipo binario g di lunghezza l nel corrispondente valore floating point x

$$x = x_{min} + \frac{x_{max} - x_{min}}{2^l - 1}(\sum_{i=1}^{l} g[i]^{l-i})$$

dove x_{min} e x_{max} sono rispettivamente il minimo e massimo valore che può assumere x e $g[i]$ è l'i-esimo allele del genotipo g.

Esempio 2.5.1. Se x è codificato con 8 bit e puo assumere valori nell'intervallo $[x_{min}, x_{max}] = [0, 1]$, allora il numero binario 01100111=103 è decodificato nel seguente valore:

$$x = 0 + \frac{1 - 0}{2^8 - 1}103 \approx 0.404$$

Valore intero	Codifica binaria	Codifica grigia
0	000	000
1	001	001
2	010	011
3	011	010
4	100	110
5	101	111
6	110	101
7	111	100

Tabella 2.1: Confronto tra codifica binaria e codifica grigia.

```
decodifica grigia (genotipo g){
      int n1=0;
      int x=0
      per i da 0 a l-1
      Se(g[i]=1){
            n1 = n1 +1
            x = x + (n1 mod 2)*2^(l-1-i)
      }
      restituisci x
}
```

Figura 2.9: Pseudoprocedura per la decodifica grigia.

Un'alternativa alla classica codifica binaria è rappresenta dalla codifica grigia (gray code). Il vantaggio di tale codifica è che a valori vicini nella rappresentazione floating point corrispondono stringhe vicine (nel senso della distanza di Hamming) nella rappresentazione binaria. Per esempio, nella codifica binaria classica, il numero 00011111=31 non è adiacente al numero 00100000=32 , nonostante i numeri 31 e 32 siano adiacenti nella rappresentazione intera. In tal modo se 32 è una soluzione migliore di 31, l'algoritmo genetico deve cambiare 6 bit per ottenerlo. La codifica grigia risolve il problema perchè interi vicini sono rappresentati da stringhe che differiscono di un solo bit. La tabella 2.1 confronta la codifica binaria classica e la codifica grigia su stringhe di 3 bit, mentre la figura 2.9 presenta una pseudoprocedura per la decodifica grigia di un genotipo.

L'operatore di crossover maggiormente utilizzato con la codifica binaria è il crossover a n punti. La differenza rispetto al crossover classico (detto a punto singolo) è l'utilizzo di n punti di taglio. Un altro operatore frequentemente uti-

lizzato con la codifica binaria è il crossover uniforme che consiste nello scambiare reciprocamente, in maniera casuale, i bit dei due genitori in ogni posizione della stringa.

L'operatore di mutazione più diffuso rimane quello proposto da Holland (paragrafo 2.3.3).

Codifica basata su numeri reali

La codifica basata su numeri floating point è quella più naturale per i problemi di ottimizzazione di parametri reali. La struttura dati utilizzata per rappresentare un individuo g è un vettore di lunghezza l dove ogni elemento $g[i]$ $(i = 1, 2, \ldots, l)$ è un numero reale che può variare in un prefissato intervallo $[-\alpha_i, \alpha_i] \subset \mathbb{R}$.

La rappresentazione basata su numeri reali non pone particolari problemi per l'operatore di crossover e quelli proposti per gli algoritmi genetici a codifica binaria possono essere riproposti senza modifiche.

Lo stesso non vale per l'operatore di mutazione. La maggior parte degli operatori di mutazione per genotipi a valori reali alterano i geni dell'individuo addizionandogli gli elementi di un vettore $\mathbf{M} = (m_1, \ldots, m_l)$

$$\mathbf{g}' = \mathbf{g} + \mathbf{M}$$

dove gli elementi di $\mathbf{M}$ possono essere generati in svariati modi, per esempio tramite distribuzioni uniformi $U(-\alpha_i, \alpha_i)$. In tal modo ogni $m_i \in \mathbf{M}$ è un valore scelto casualmente nell'intervallo $[-\alpha_i, \alpha_i]$ con eguale probabilità.

Codifica ad albero

La codifica ad albero è utilizzata soprattutto nell'evoluzione di programmi, dove, piuttosto che generare direttamente soluzioni di un dato problema di ottimizzazione, l'obbiettivo è quello di derivare algoritmi in grado di risolvere particolari problemi computazionali. Tale branca di applicazione degli AG è chiamata Programmazione Genetica (PG) [104].

La struttura dati utilizzata è un albero con nodi terminali, detti foglie, e nodi non terminali. I nodi terminali, dai quali discendono sottoalberi, possono essere costanti e variabili specifiche del problema, mentre i nodi non terminali possono essere funzioni come +, -, *, / e $\sqrt{}$ o strutture di controllo del tipo *if then else*. La figura 2.10 rappresenta l'albero di un algoritmo che calcola l'espressione $\sqrt{A^3}$. L'albero si legge dal basso verso l'alto: prima si esegue il prodotto A*A, che si moltiplica ancora per A e, infine, se ne calcola la radice quadrata.

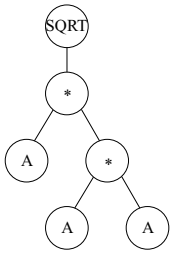


Figura 2.10: Esempio di codifica ad albero di un algoritmo che calcola l'espressione $\sqrt{A^3}$.

L'operatore di crossover e di mutazione presentano alcune differenze rispetto ai corrispondenti operatori degli AG.

L'incrocio consiste nella scelta di due punti di taglio, uno per ogni genitore, e nello scambio dei sottoalberi discendenti. La figura 2.11 mostra un esempio di crossover tra due alberi. Come si evince dalla figura 2.11, per effetto dell'incrocio, le dimensioni dei programmi possono aumentare o diminuire.

L'operatore di mutazione, quando viene applicato, sostituisce sottoalberi con nuovi sottoalberi generati casualmente. Anche in questo caso le dimensioni dei programmi possono aumentare o diminuire.

2.5.2 Metodi di selezione

La selezione è uno dei processi fondamentali di un algoritmo genetico poichè elimina gli individui con fitness bassa e crea una o più copie di individui con fitness alta da cui discendono gli individui della nuova popolazione.

La selezione incide in maniera significativa sulla dinamica dell'algoritmo genetico: una pressione selettiva troppo forte può dar luogo alla predominanza di qualche individuo con fitness particolarmente alta e condurre l'algoritmo genetico in un ottimo locale da cui difficilmente riuscirà a uscire; d'altro canto, una pressione selettiva troppo debole può dar luogo a un eccessivo aumento del tempo d'esecuzione necessario a trovare una soluzione accettabile.

Gli operatori di selezione possono rimpiazzare l'intera popolazione (in tal caso si parla di AG generazionali) o soltanto una parte di essa (AG steady state). Inoltre, gli operatori di selezione possono scegliere più volte o soltanto una volta uno stesso individuo per la riproduzione. Nel primo caso si parla di operatori con rimpiazzamento (with replacement), nel senso che l'individuo selezionato per la riproduzione, dopo l'accoppiamento, viene reinserito nella vecchia popolazione e

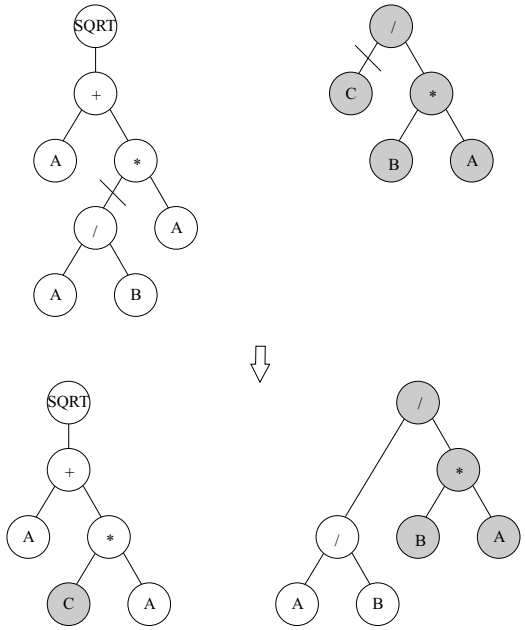


Figura 2.11: Esempio di crossover tra due alberi. Viene scelto casualmente un punto di taglio sui due alberi e i sottoalberi discendenti vengono scambiati.

può essere selezionato nuovamente. Nel secondo caso si parla di operatori senza rimpiazzamento (without replacement).

Sia negli AG generazionali che negli steady-state può succedere che l'individuo migliore venga perso nel passaggio alla generazione successiva. I modelli che garantiscono la sopravvivenza del miglior individuo sono detti elitistici, o k-elitistici se garantiscono la sopravvivenza dei migliori k.

Di seguito sono presentati alcuni tra i metodi di selezione maggiormente utilizzati nelle applicazioni.

Selezione proporzionale

L'originario modello di Holland impiegava la selezione proporzionale alla fitness (eq. 2.1) utilizzando uno schema con rimpiazzamento. Tuttavia, tale schema presenta una pressione selettiva molto alta: gli individui con fitness alta e i loro discendenti si moltiplicano troppo velocemente (eq. 2.3) e impediscono di fatto all'algoritmo genetico di effettuare ulteriori significative esplorazioni [126]. L'eccessiva pressione selettiva può essere comunque mitigata utilizzando algoritmi genetici steady state con uno schema senza rimpiazzamento.

Forse per motivi storici, insieme alla selezione a torneo introdotta per diminuire la pressione selettiva, la selezione proporzionale è tra le più utilizzate nelle applicazioni pratiche.

Selezione di Boltzmann

La selezione di Boltzmann [73] permette una pressione selettiva variabile durante l'evoluzione dell'algoritmo genetico.

Inizialmente una pressione selettiva bassa consente agli individui meno idonei di riprodursi quasi quanto quelli più idonei. Questo permette di mantenere una significativa diversità nella popolazione e un buon campionamento dello spazio delle soluzioni. Successivamente, l'aumento della pressione selettiva durante l'evoluzione dell'algoritmo genetico favorisce la riproduzione degli individui più idonei. Se nella fase iniziale, caratterizzata da bassa pressione selettiva, l'algoritmo genetico riesce a individuare la zona giusta dello spazio di ricerca, l'aumento della pressione selettiva guida la ricerca proprio in quella zona favorendo la convergenza verso una buona soluzione.

Una tipica applicazione della selezione di Boltzmann consiste nell'assegnare a ogni individuo g_i un valore atteso

$$ExpVal(g_i, t) = \frac{e^{f_i/T}}{\langle e^{f_i/T} \rangle_t} \tag{2.8}$$

dove T è la variabile "temperatura" che, diminuendo gradualmente all'aumentare della generazione t, garantisce l'aumento della pressione selettiva. Il simbolo $\langle\rangle_t$ indica la media sulla popolazione alla generazione t.

Gli effetti della selezione di Boltzmann su algoritmi genetici con popolazione finita sono stati studiati da Prügel-Bennett e Shapiro [150, 151].

Selezione in base al rango

La selezione in base al rango riduce significativamente la pressione selettiva rispetto alla selezione proporzionale [126]. Gli individui della popolazione sono classificati in base alla fitness e i relativi valori attesi dipendono dalla posizione (rango) che occupano nella classifica. Nel metodo di classificazione lineare proposto da Baker [15], ogni individuo è classificato in ordine crescente di idoneità, da 1 a n. L'utente scegli il valore atteso $Max \geq 0$ dell'individuo di rango n e il valore atteso di ogni individuo g_i della popolazione alla generazione t è

$$ExpVal(g_i, t) = Min + (Max - Min)\frac{rango(g_i, t) - 1}{n - 1} \tag{2.9}$$

dove Min è il valore atteso dell'individuo di rango 1. Dati i vincoli $Max \geq 0$ e $\sum_{i=1}^{n} ExpVal(g_i, t) = n$, affinchè la consistenza numerica della popolazione si mantenga inalterata tra una generazione e l'altra, è necessario che $1 \leq Max \leq 2$ e $Min = 2 - Max$. Baker suggerì di porre Max=1.1 e mostrò che questo schema dava risultati migliori di quello proporzionale all'idoneità in alcune applicazioni.

Gli effetti della selezione in base al rango sono stati studiati da Blickle [23] e da Prügel-Bennett e Rogers [149, 155].

Selezione a torneo

Diversamente dalle precedenti, la selezione a torneo non richiede operazioni globali sulla popolazione e risulta più efficiente in termini di tempo di calcolo [126].

Nel tipo più comune di selezione a torneo, si scelgono a caso due individui e si genera un numero casuale $c \in [0, 1]$. Se c risulta minore di un parametro $k \in [0, 1]$ fissato, per esempio $k = 0.75$, allora si seleziona il più idoneo, altrimenti si sceglie il meno idoneo. Se si applica uno schema con rimpiazzamento, inoltre, i due individui sono reinseriti nella vecchia popolazione e possono essere scelti di nuovo.

Per quanto riguarda la pressione selettiva, la selezione a torneo è equivalente a quella in base al rango [126]. Per un confronto teorico si veda [149].

2.6 Alcune applicazioni degli Algoritmi Genetici

Gli AG sono stati utilizzati per risolvere un gran numero di problemi scientifici e ingegneristici in molteplici settori d'applicazione quali la Programmazione Automatica, la Teoria dei Giochi, le Scienze Cognitive, la Robotica, la Teoria dei Giochi, la Biologia Molecolare, l'Ottimizzazione e molti altri.

Pur non garantendo la convergenza alla soluzione ottimale, generalmente gli AG sono in grado di trovare soluzioni sufficientemente buone in tempi abbastanza rapidi. Le tecniche specializzate per risolvere problemi specifici, quando esistono, quasi sempre hanno prestazioni superiori agli AG. Dunque, i contesti d'applicazione migliori per gli algoritmi genetici sono costituiti da quei problemi per i quali le tecniche di risoluzione specifiche richiedono tempi di calcolo troppo lunghi o dove tali tecniche non esistono affatto. Infine, un terzo e interessante contesto d'applicazione è quello in cui gli AG sono utilizzati come modelli scientifici dell'evoluzione biologica.

Nei paragrafi successivi sono presentati alcuni lavori in cui gli AG sono stati applicati nei tre contesti sopra citati. In particolare, nel paragrafo 2.6.1 è introdotta

l'applicazione degli AG a problemi d'ottimizzazione combinatoria, nel paragrafo 2.6.2 sono presentati alcuni lavori sull'evoluzione di strategie nel gioco del Dilemma del Prigioniero, nel paragrafo 2.6.3 sono descritti alcuni lavori sulle relazioni tra evoluzione e apprendimento, mentre nel paragrafo 2.6.4 è presentata sinteticamente la Robotica Evolutiva con alcune recenti applicazioni. Altre applicazioni degli Algoritmi Genetici sono illustrate nel capitolo successivo che introduce gli Automi Cellulari.

2.6.1 Ottimizzazione

Negli ultimi anni un numero sempre crescente di ricerche hanno visto l'applicazione degli algoritmi genetici a problemi complessi d'ottimizzazione combinatoria [28, 3, 164, 114, 145]. Tali problemi, a parte qualche caso teorico particolare, appartengono alla classe dei problemi NP-Completi per i quali non è noto alcun algoritmo deterministico in grado di risolverli in tempo polinomiale. Questo vuol dire che al crescere delle dimensioni del problema, il tempi d'esecuzione possono crescere in maniera esponenziale anche se si utlizza l'algoritmo noto più efficiente, rendendo di fatto impossibile trattare problemi di dimensioni elevate. Il successo degli AG nel campo dell'ottimizzazione combinatoria è dovuto proprio alle applicazioni a problemi di grosse dimensioni. Gli AG, infatti, pur non garantendo la convergenza alla soluzione ottimale, solitamente riescono a trovare soluzioni sufficientemente buone in tempi brevi, rappresentando un buon compromesso tra qualità delle soluzioni e rapidità d'esecuzione. Di seguito è brevemente illustrata una possibile metodologia d'applicazione degli AG a problemi d'ottimizzazione combinatoria considerando come caso rappresentativo il Problema del Commesso Viaggiatore (TSP – Traveller Salesman Problem).

Il Problema del Commesso Viaggiatore può essere descritto nel seguente modo: date n città e la distanza tra ognuna di esse, l'obbiettivo del commesso viaggiatore è visitare ognuna delle n città una sola volta minimizzando il commino da percorrere. In termini formali il TSP può essere descritto come un grafo non direzionale $G = \langle V, E \rangle$, dove $V = \{v_0, v_1, \ldots, v_{n-1}\}$ è l'insieme degli n vertici rappresntanti le città da visitare, E è l'insieme degli $n(n-1)/2$ archi, e l'obbiettivo è quello di trovare un ciclo Hamiltoniano di lunghezza minima. Si noti che per n città esistono $n!$ cicli Hamiltoniani (ogni ciclo è una permutazione sugli elementi di V) e che una ricerca esaustiva è praticamente improponibile per valori di n molto grandi. La funzione

di fitness per il TSP può essere così definita:

$$f_{TSP}(hc_k) = \sum_{i=0}^{n-1} d(v_i^k, v_{|i+1|_n}^k)$$

in cui $hc_k = \{v_0^{(k)}, v_1^{(k)}, \ldots, v_{n-1}^{(k)}\}$ è il k-esimo degli $n!$ cicli Hamiltoniani in G, $|\ldots|_n$ rappresenta l'operatore mod n e $d(v_i^k, v_{|i+1|_n}^k)$ la distanza tra due nodi successivi. In questo modo la funzione di fitness calcola la lunghezza del commino del ciclo Hamiltoniano considerato.

Il modo più naturale per rappresentare un ciclo è utilizzare una codifica a numeri interi. Per esempio, considerando un problema di dimensione $n = 10$, una possibile rappresentazione di un ciclo può essere la seguente: $hc_1 = 6874391205$. Secondo tale rappresentazione il commesso viaggiatore, partendo dalla città 6, deve visitare nell'ordine la città 8, la 7, la 4, e così via, per ritornare, infine, alla città di partenza. Come risulta subito evidente, l'operatore di crossover classico ha altissime probabilità di generare cicli non significativi. Per esempio, se l'individui hc_1 si ricombina con l'individuo $hc_2 = 2804391657$, con punto di taglio nella posizione 2, si ottengono i seguenti discendenti:

$$\begin{array}{l} hc_1 = 68|74391205 \\ hc_2 = 28|04391657 \end{array} \Rightarrow \begin{array}{l} hc_1' = 6804391657 \\ hc_2' = 2874391205 \end{array}$$

e nessuno dei due è un ciclo Hamiltoniano valido. Infatti l'individuo hc_1' non prevede il passaggio dalla città 2, mentre l'individuo hc_2' non prevede il passaggio dalla città 6. Possibili soluzioni a questo problema sono l'utilizzo di algoritmi riparatori che generino percorsi corretti oppure l'applicazione di operatori di crossover non tradizionali. Un operatore di crossover non tradizionale potrebbe, per esempio, copiare dal primo genitore la sequenza delle città dall'inizio fino al punto di taglio aggiungendo le rimanenti dal secondo genitore, cominciando anche in questo caso dall'inizio e copiando solo le città non precedentemente scelte. Per generare il secondo discendente basta invertire il ruolo dei due genitori, come nell'esempio seguente:

$$\begin{array}{l} hc_1 = 68|74391205 \\ hc_2 = 28|04391657 \end{array} \Rightarrow \begin{array}{l} hc_1' = 6820439157 \\ hc_2' = 2867439105 \end{array}$$

Per quanto riguarda l'operatore di mutazione quello più utilizzato consiste nell'effettuare uno scambio tra due o più geni scelti a caso. Nell'esempio seguente il

	Il giocatore A coopera	Il giocatore A tradisce
Il giocatore B coopera	3,3	5,0
Il giocatore B tradisce	0,5	1,1

Tabella 2.2: Matrice del punteggio per il Dilemma del Prigioniero. I numeri separati da virgola sono, rispettivamente, i punti dei giocatori A e B.

secondo e il quinto gene dell'individuo hc'_1 sono scambiati generando l'individuo hc''_1

$$hc'_1 = 6\mathbf{8}20\mathbf{4}39157 \Rightarrow hc''_1 = 64\mathbf{2}0\mathbf{8}39157$$

Per una rassegna dettagliata delle codifiche e degli operatori maggiormente utilizzati in letteratura si veda il lavoro di Larrañaga & al. [113].

2.6.2 Evoluzione di strategie: il Dilemma del Prigioniero

Il Dilemma del Prigioniro è un semplice gioco proposto da Merrill Flood e Melvin Dresher negli anni '50 come parte delle ricerche sulla teoria dei giochi promosse dalla Rand Corporation per le possibili applicazioni sulla corsa agli armamenti [11].

Due persone, A e B, sono state arrestate e rinchiuse in celle separate in modo che non possano comunicare tra loro. Sia A che B possono, tuttavia, essere incriminati solo per un reato minore per il quale sarebbero condannati a 2 anni di prigione. A ognuno viene proposto un accordo: se tradisce e accetta di testimoniare contro l'altro avrà abbonata la pena mentre l'altro sarà condannato a 5 anni. Se entrambi tradiscono, però, saranno condannati tutti e due a 4 anni per ammissione di colpa mentre se cooperano saranno condannati a 2 anni per il reato minore. Una partita consiste in una serie di mosse in cui ogni giocatore decide, indipendentemente, se cooperare o tradire. I punteggi delle singole mosse, riportati in tabella 2.2, coincidono con il numero di anni risparmiati per ogni giocatore.

Il Dilemma del Prigioniero è stato studiato ampiamente da Robert Axelrod [12, 13] dell'Università del Michigan che organizzò, negli anni '80, veri e propri tornei in cui vari algoritmi di strategie erano messi in competizione. Ogni algoritmo ricordava la decisione (collaborazione o tradimento), sua e dell'avversario, delle tre mosse precedenti giocando un numero prefissato di partite con ogni altro programma. La strategia vincente risultò la tit for tat (occhio per occhio) che cooperava nella prima mossa e replicava le decisioni dell'avversario nelle mosse successive. Axelrod volle verificare se un algoritmo genetico fosse in grado di far

Caso	Configurazione	Mossa
0	00 00 00	$g[0] \in \{0, 1\}$
1	00 00 01	$g[1] \in \{0, 1\}$
2	00 00 10	$g[2] \in \{0, 1\}$
3	00 00 11	$g[3] \in \{0, 1\}$
4	00 01 00	$g[4] \in \{0, 1\}$
...	...	...
63	11 11 11	$g[63] \in \{0, 1\}$

Tabella 2.3: Schema per la codifica del genotipo del Dilemma del Prigioniero. Ogni bit può assumere il valore 0 (cooperazione) o 1 (tradimento) a seconda della configurazione delle tre mosse precedenti. Per esempio se $g[0] = 1$ vuol dire che la strategia risponde tradendo se nelle precedenti 3 mosse entrambi i giocatori hanno cooperato.

evolvere strategie per giocare con successo al Dilemma del Prigioniero. Utilizzò un algoritmo genetico con codifica binaria in cui i primi 64 bit servivano per codificare la strategia. Infatti ogni strategia, potendo ricordare le tre mosse precedenti (sue e dell'avversario), ha 64 possibili configurazioni da gestire (tabella 2.3). Axelrod impiegò, però, stringhe di 70 bit nelle quali i 6 bit in più codificavano ipotetiche partite precedenti utilizzate dalla strategia per decidere come giocare nelle prime mosse. Nel primo esperimento venne utilizzata una popolazione di 20 strategie generate casualmente. Ognuna giocava ripetutamente contro 8 strategie fisse scelte da Axelrod come "ambiente" cui gli individui dell'algoritmo genetico dovevano adattarsi. Il valore di fitness di un genotipo era, infatti, il punteggio medio ottenuto giocando contro il campione delle 8 strategie fisse. Axelrod eseguì 40 simulazioni da 50 generazioni. La maggior parte delle strategie evolute risultarono simili al tit for tat ma alcune ottennero risultati nettamente superiori. Questo è un risultato notevole, specialmente a fronte del fatto che l'algoritmo genetico si è mosso in uno spazio di ricerca di 2^{70} possibili soluzioni. Tuttavia Axelrod ipotizzò che tale risultato fosse dovuto alla staticità dell'ambiente che non permetteva all'algoritmo genetico di evolvere strategie generali, ma solo strategie particolari che funzionavano bene sul campione delle 8 strategie fisse. Le strategie evolute, infatti, potevano dare scarsi risultati se messe in competizione con strategie diverse da quelle del campione fisso. Per studiare gli effetti di un ambiente variabile Axelrod fece altri esperimenti in un contesto di co-evoluzione dove l'idoneità di un indi-

viduo era calcolata facendolo giocare più volte contro ogni altro individuo della popolazione, incluso se stesso. Così l'abmiente cambiava di generazione in generazione poichè gli avversari stessi si evolvevano. In questo secondo caso Axelrod osservò che inizialmente l'algoritmo genetico evolveva strategie non cooperative, ma dopo 10 o 20 generazioni la tendenza si invertiva evolvendo strategie che premiavano la cooperazione e punivano il tradimento, cioè strategie simili a tit for tat. Tali strategie si sono dimostrate decisamente più robuste di quelle evolute in ambiente statico nel senso che hanno dato buoni risultati sia contro strategie simili che contro strategie non cooperative.

Paul G. Harrald e David B. Fogel [81] hanno utilizzato la Programmazione Genetica [104] per evolvere reti neurali di tipo feed-forward rappresentanti strategie del Dilemma del Prigioniero capaci di esibire un continuum di comportamenti anzichè le sole due opzioni di collaborazione e tradimento. I risultati sperimentali hanno mostrato che l'evoluzione di strategie cooperative può emergere, anche se in maniera non stabile durante l'evoluzione, a fronte di un "minimo" di complessità nell'architettura della rete.

Jae C. Oh [143] si è interessato dei meccanismi in grado di promuovere la cooperazione nell'evoluzione di strategie del Dilemma del Prigioniero eseguendo alcuni esperimenti in cui gli individui erano incoraggiati a cooperare con individui simili (a livello genotipico) e tradire individui dissimili. Più precisamente, individui cooperativi erano incoraggiati a cooperare con individui cooperativi e a tradire individui dissimili. I risultati del lavoro di Oh hanno mostrato un significativo aumento del livello di cooperazione in ogni esperimento eseguito con questo meccanismo.

Gli studi sul Dilemma del Prigioniero hanno, inoltre, ispirato ricerche sull'emergenza della comunicazione. Michael Oliphant [144], del Dipartimento di Scienze Cognitive dell'Università di San Diego, ha utilizzato un algoritmo genetico per studiare le condizioni neccessarie all'evoluzione di un sistema di comunicazione di Saussurean, cioè di un sistema in cui la comunicazione si basa sulla condivisione di un linguaggio non ambiguo tra tutti gli individui di una popolazione. Oliphant ha dimostrato che, anche in assenza di pressione selettiva sulle abilità di comunicazione, un sistema di Saussurean può essere evoluto, purchè gli individui della popolazione interagiscano (comunichino) più volte con gli stessi partners e abbiano la capacità di diversificare le risposte in base all'esperienza maturata nelle precedenti interazioni.

Molti altri lavori sono stati prodotti sull'evoluzione delle strategie del Dilemma del Prigioniero attraverso Algoritmi Genetici. Si vedano, per esempio, i lavori di Crowley & al. [42], di Ashlock & al. [6] e di Hoffman [85].

2.6.3 Evoluzione e apprendimento

L'evoluzione e l'apprendimento sono due forme di adattamento biologico agenti a differenti scale spazio-temporali: la prima su popolazioni di individui geograficamente distribuiti in un ambiente su scala temporale generazionale; il secondo sui singoli individui nell'arco della vita degli stessi.

La nota ipotesi di Lamark, secondo la quale i caratteri acquisiti nel corso della vita di un organismo (per esempio le modificazioni dei pesi sinaptici per effetto dell'apprendimento) possono essere trasmessi geneticamente ai suoi discendenti, è stata rifiutata, a causa delle innumerevoli prove a suo sfavore, dalla quasi totalità della comunità scientifica [126]. Tuttavia, benchè i cambiamenti nei tratti fenotipici non possano essere codificati a livello genotipico, Baldwin [16] e Waddington [180] hanno suggerito l'ipotesi secondo la quale l'apprendimento può influenzare l'evoluzione in maniera indiretta ma significativa. Baldwin, in un lavoro del 1896, fece notare che, se l'apprendimento aiuta a sopravvivere, gli individui capaci d'imparare avranno un maggior numero di discendenti, aumentando così il numero di geni responsabili dell'apprendimento. E, se l'ambiente rimane relativamente stabile, ciò può condurre, tramite la selezione, alla codifica genetica di un carattere che inizialmente doveva essere appreso. Baldwin chiamò questo meccanismo "selezione organica", ma esso fu in seguito ribattezzato da Simpson [166] "effetto Baldwin". Waddington propose un meccanismo simile, l'"assimilazione genetica". Secondo Waddington, se degli organismi sono soggetti a sconvolgimenti ambientali, talvolta riescono ad adattarsi nel corso della vita poichè riescono ad acquisire nuovi tratti fisici o comportamentali. Se i geni relativi ai tratti acquisiti sono presenti nei genotipi degli individui della popolazione, anche se non espressi, essi possono esprimersi abbastanza rapidamente, specie quando gli adattamenti fenotipici sono quelli che evitano l'estinzione della specie. In entrambi i casi, tuttavia, non era chiaro quanto l'effetto dell'assimilazione genetica dei tratti appresi avvenisse frequentemente e quanto fosse importante nella dinamica del processo evolutivo.

Nel tentativo di rispondere a tali questioni, Hinton e Nowlan [84] proposero un modello in cui un algoritmo genetico giocava il ruolo dell'evoluzione su una popolazione di reti neurali, capaci d'apprendimento, ognuna con 20 possibili connessioni. Ogni connessione poteva assumere uno tra i valori fissi 0 e 1, oppure il valore ?, che poteva diventare 0 o 1 nel corso dell'apprendimento. L'obbiettivo era quello di evolvere un individuo che avesse tutte le connessioni fisse e uguali a quelle di una configurazione prefissata. Tutte le reti con valori esatti nelle posizioni fisse e simboli ? nelle altre avevano la capacità d'imparare la giusta configurazione attraverso un

algoritmo d'apprendimento per tentativi casuali, dove ogni tentativo consisteva nell'assegnare alle posizioni con il simbolo ? un simbolo scelto a caso tra i due simboli fissi 0 e 1. Nell'algoritmo genetico, ogni rete neurale era rappresentata attraverso una stringa di lunghezza 20 dove ogni gene poteva assumere uno dei tre simboli 0, 1 oppure ?. La popolazione iniziale era composta da 1000 individui generati casualmente aventi in media il 25% di simboli 1, 25% di simboli 0 e il 50% di simboli ?. A ogni generazione ogni individuo g eseguiva 1000 tentativi d'apprendimento e l'idoneità era calcolata secondo la legge $f = f(g) = 1 + 19(1000 - t)/1000 \in [1, 20]$, dove $t \in \{0, 1, \ldots, 1000\}$ era il numero di tentativi (trials) necessari a individuare la configurazione giusta. I cromosomi degli individui della popolazione non venivano modificati per effetto dell'apprendimento e i genitori trasmettevano ai propri discendenti i loro alleli originali. Tuttavia, gli individui più abili nell'apprendimento, essendo quelli con fitness più alta, venivano premiati con un maggior numero di discendenti alla generazione successiva. Alcuni esperimenti [19, 82, 67], mostrarono che senza apprendimento (genotipi interamente formati con i soli simboli 0 e 1) l'idoneità media non aumentava al trascorrere del tempo, mentre questo succedeva nelle prove con apprendimento, anche senza la trasmissione genetica delle abilità acquisite. Hinton e Nowlan interpretarono il fenomeno come conseguenza dell'effetto Baldwin: gli individui in grado di apprendere rapidamente le giuste connessioni tendevano e essere scelti per la riproduzione e gli incroci tra questi individui tendevano ad aumentare il numero di alleli fissati correttamente, migliorando le capacità d'apprendimento dei discendenti.

Una critica al lavoro di Hinton e Nowlan fu relativa all'assenza di un fenotipo. Akley e Littman [1] proposero un modello più plausibile da questo punto di vista, basato su una griglia bidimensionale popolata da predatori, cibo e agenti, questi ultimi in grado d'apprendere e d'evolvere. Ogni agente era costituito da due reti neurali. Una rete valutava lo stato dell'agente (livello d'energia, distanza dal cibo, da un altro agente o da un predatore, ecc.), mentre una seconda rete determinava l'azione dell'agente (stare fermo o muoversi in una delle quattro celle adiacenti) sulla base della valutazione della prima rete. I pesi della rete di valutazione erano fissati alla nascita mentre quelli della rete d'azione potevano variare nel corso della vita attraverso un algoritmo d'apprendimento. I genotipi codificavano gli 84 pesi delle due reti neurali utilizzando 4 bit per ognuno, per un totale di 336 bit. In tal modo, anche i pesi della rete d'azione venivano codificati ma, al livello genotipico essi non variavano anche se, per effetto dell'algoritmo d'apprendimento, gli stessi potevano variare nella rete neurale. Superata una certa soglia d'energia, gli agenti potevano riprodursi per clonazione, generando un discendente con il cromosoma del

genitore variato per mutazione, oppure per crossover, ricombinando i cromosomi di due agenti adiacenti nella griglia. L'idoneità non era calcolata esplicitamente, ma emergeva dal comportamento dell'agente nell'ambiente: gli agenti più idonei riuscivano più degli altri a evitare i predatori, a procurarsi il cibo e, ovviamente, a riprodursi. Nei test eseguiti gli esperimenti in cui agivano insieme evoluzione e apprendimento hanno dato risultati migliori, nel senso che le popolazioni di agenti si estinguevano meno rapidamente, rispetto agli esperimenti eseguiti con il solo apprendimento o con la sola evoluzione, suffragando l'ipotesi che l'apprendimento può influenzare significativamente l'evoluzione.

Nolfi, Elman e Parisi [137, 148, 136] hanno dimostrato che l'evoluzione e l'apprendimento possono influenzarsi reciprocamente anche quando gli obbiettivi dell'apprendimento e dell'evoluzione sono differenti. In una serie d'esperimenti i tre ricercatori hanno fatto evolvere una popolazione di agenti valutando la fitness sulla base di un prefissato compito (cercare cibo distribuito casualmente su un reticolo bidimensionale) richiedendo, allo stesso tempo, che ogni agente apprendesse un ulteriore compito (predire le conseguenze dell'attivazione di alcuni neuroni della rete neurale, utilizzata come sistema di controllo dell'agente). Gli autori hanno mostrato che, dopo poche generazioni, gli individui che avevano acquisito il compito d'apprendimento risultavano anche più abili nella ricerca del cibo, spiegando il fenomeno come risultato dell'interazione tra evoluzione e apprendimento. Più precisamente, essi hanno proposto che la chiave di lettura potrebbe essere che l'evoluzione tende a selezionare quegli individui che acquisendo il compito d'apprendimento, contestualmente diventano più abili anche nella ricerca del cibo. I risultati di altri esperimenti in cui agli individui non era richiesto di acquisire il compito d'apprendimento hanno prodotto, infatti, risultati decisamente inferiori.

Munroe e Cangelosi [130] hanno utilizzato un modello di vita artificiale per studiare gli effetti del costo dell'apprendimento e della variazione culturale nell'effetto Baldwin per l'origine e l'evoluzione del linguaggio dimostrando che, se il costo dell'apprendimento è elevato, gli agenti possono acquisire geneticamente alcune caratteristiche esplicite del linguaggio cui sono esposti. Inoltre, quando le strutture del linguaggio variano durante la trasmissione culturale, il processo baldwiniano può determinare l'assimilazione di una predisposizione all'apprendimento piuttosto che l'assimilazione di particolari strutture di uno specifico linguaggio.

Numerosi altri studi sono stati condotti sull'interazione tra evoluzione e apprendimento, tra cui alcune applicazioni all'ottimizzazione di funzioni [182], a problemi d'ottimizzazione combinatoria [159], all'addestramento di reti neurali [36, 105].

2.6.4 Robotica Evolutiva

La Robotica Evolutiva [138] è una branca della Robotica che si occupa della progettazione e realizzazione di robot autonomi i quali acquisiscono le proprie abilità attraverso la sola interazione con l'ambiente durante un processo evolutivo simulato tramite Algoritmi Genetici. Una popolazione iniziale di genotipi, ognuno dei quali codifica il sistema di controllo e talvolta anche la morfologia del robot, viene generata in maniera casuale e collocata in un ambiente. L'interazione del robot con l'ambiente determina la valutazione delle performance dello stesso in relazione a un prefissato compito. I migliori individui sono selezionati per la riproduzione generando discendenti i cui genotipi sono ottenuti tramite l'applicazione dei classici operatori di ricombinazione e/o mutazione. Il processo viene iterato per un certo numero di generazioni fino a quando viene generato un particolare robot le cui performance soddisfino un prefissato criterio.

Le attuali ricerche in Robotica Evolutiva spesso riguardano agenti reattivi [64, 135, 139, 142], in cui l'azione dei motori è basata principalmente sullo stato corrente dei sensori, o agenti guidati dalla propria dinamica interna [18, 95, 74], in cui l'azione dei motori è determinata principalmente dalla dinamica interna del proprio sistema di controllo. Nolfi e Marocco [140] hanno, tuttavia, mostrato che robot capaci d'integrare informazioni senso-motorie nel tempo possono superare significativamente in prestazioni robot di tipo reattivo. I due ricercatori hanno utilizzato come sistema di controllo una semplice rete neurale di tipo feed-forward a due strati aggiungendone un terzo intermedio costituito da due neuroni che ricevono connessioni dai neuroni di input (sensori) e da essi stessi e proiettano connessioni verso i neuroni di output (motori). In tal modo l'attivazione dei motori dipende ancora dall'attivazione dei neuroni di input ma anche dall'attivazione, dipendente dalle informazioni raccolte nelle precedenti interazioni del robot con l'ambiente, dei due neuroni dello strato intermedio. Gli esperimenti eseguiti hanno replicato due precedenti esperimenti di navigazione in un ambiente rettangolare [64, 139]. Nel primo caso al robot, posizionato in maniera casuale nell'ambiente, era richiesto di navigare verso uno di due angoli opposti (nord-ovest e sud-est per l'esattezza), mentre nel secondo caso al robot, posizionato in maniera casuale sulla linea di equidistanza tra due cilindri collocati nell'ambiente, era richiesto di navigare verso il cilindro più grande. Per ognuno dei due casi sono state eseguite 10 esecuzioni di un algoritmo genetico con una popolazione di 100 individui per far evolvere sia i sistemi di controllo di tipo senso-motorio, sia i sistemi di controllo con i due neuroni intermedi. Il genotipo binario dell'algoritmo genetico adottato codificava

i pesi sinaptici delle reti neurali (8 di input e 2 di output per il primo caso; 8 di input, 2 intermedi e 2 di output per il secondo caso) utilizzando 8 bit per ognuno. Inoltre la popolazione alla generazione $t+1$ era ottenuta scegliendo i migliori 20 individui della popolazione al tempo t e generando per ognuno 5 discendenti tramite mutazione. I risultati ottenuti hanno mostrato che i robot dotati del nuovo sistema di controllo hanno superato significativamente in prestazioni i robot con sistema di controllo senso-motorio, ma le strategie evolute non si sono discostate di molto da quelle evolute nel caso di robot senso-motori. I due ricercatori hanno spiegato il fenomeno ipotizzando che i due neuroni interni sono stati utilizzati dall'agente per modulare il comportamento senso-motorio base sulle precedenti interazioni con l'ambiente. Infatti, inibendo l'azione dei due neuroni intermedi, si è osservato, in entrambi gli esperimenti, lo stesso comportamento di quello osservato per gli agenti senso-motori.

Recentemente Marocco, Cangelosi e Nolfi [122] hanno utilizzato le tecniche e le metodologie della Robotica Evolutiva per lo studio dell'evoluzione del linguaggio. Il lavoro si basa su un precedente studio di Nolfi e Marocco [141] sulla categorizzazione di oggetti di forme differenti sulla base di informazioni tattili tramite agenti dotati di sistemi di controllo di tipo senso-motorio. Nel nuovo modello gli agenti robotici condividono la categorizzazione degli oggetti tramite l'invio di un segnale (nome dell'oggetto categorizzato) definito dallo stato di alcuni neuroni di output. L'ambiente consiste in uno spazio tridimensionale all'interno del quale è collocato un oggetto da categorizzare (un cubo o una sfera). L'agente è un braccio costituito da tre segmenti, controllato da una rete neurale di tipo feed-forward a tre strati con 11 neuroni di input, 3 nascosti e 8 di output. I primi 9 neuroni di input codificano la posizione del braccio nell'ambiente mentre gli ultimi 2 ricevono l'input (segnale) da qualche altro agente. Similmente, i primi 6 neuroni di output regolano l'azione dei motori mentre gli ultimi 2 codificano il segnale da comunicare agli altri agenti. Gli agenti, evoluti tramite un algoritmo genetico il cui genotipo binario codifica i pesi sinaptici della rete neurale, sono selezionati in funzione della propria abilità nel categorizzare gli oggetti e non sulla base delle abilità linguistiche. La popolazione dell'algoritmo genetico è costituita da 80 individui dove i migliori 20 sono fatti riprodurre generando ognuno 4 discendenti per mutazione. La popolazione al tempo $t-1$ dei genitori comunica i segnali linguistici ai discendenti della popolazione al tempo t. Questi ultimi sono sottoposti al test di fitness e, dopo la riproduzione, assumono il ruolo di comunicatori verso i nuovi discendenti. Sono stati eseguiti quattro esperimenti: nel primo (Parent-From0) i discendenti possono ricevere segnali solo dai genitori sin dall'inizio della simulazione, nel secondo (Parent-From50) i

	From0	From50
Parent	5 (27%, 75%)	7 (63%, 100%)
All	0 (7%, 20%)	0 (5%, 27%)

Tabella 2.4: Risultati degli esperimenti sull'emergenza della comunicazione. Il primo valore rappresenta il numero delle popolazioni in cui è emersa comunicazione mentre i numeri tra parentesi rappresentano le percentuali medie dei migliori comunicatori sulle 10 replicazioni dell'esperimento e sulla migliore popolazione evoluta rispettivamente.

discendenti possono ricevere segnali solo dai genitori dalla generazione 50 in poi, nel terzo (All-From0) i discendenti possono ricevere segnali da tutti gli individui della popolazione precedente sin dall'inizio della simulazione, nel quarto (All-From50) i discendenti possono ricevere segnali da tutti gli individui della popolazione precedente dalla generazione 50 in poi. Per ognuno dei quattro casi sono state eseguite 10 simulazioni dell'algoritmo genetico, ognuna con popolazione iniziale differente. I risultati, illustrati in tabella 2.4, evidenziano che l'emergenza della comunicazione è favorita da una prima fase (le prime 50 generazioni) caratterizzata dall'assenza di segnali e da una seconda in cui lo scambio di segnali avviene solo tra genitori e discendenti diretti; quando la comunicazione è introdotta dall'inizio e avviene solo tra genitori e discendenti diretti, le probabilità di ottenere agenti caratterizzati sia da un buon comportamento che da una buona comprensione e trasmissione dei segnali diminuisce sensibilmente (da 7 a 5 popolazioni su dieci); negli ultimi due casi non emerge alcuna forma di comunicazione stabile. L'emergenza del linguaggio comporta, inoltre, diretti benefici sulle abilità degli agenti. Infatti le popolazioni in cui emerge il linguaggio superano significativamente, in termini di fitness, le popolazioni dove questo non avviene.

2.7 Discussione

Gli Algoritmi Genetici sono algoritmi iterativi ispirati ai meccanismi della selezione naturale e della riproduzione sessuale, ampiamente studiati dal punto di vista teorico e applicati sia come algoritmi d'ottimizzazione, sia come modelli scientifici dell'evoluzione biologica.

Come illustrato negli esempi applicativi, definita un'opportuna codifica delle soluzioni candidate, gli AG possono trovare buone soluzioni anche in spazi di ricer-

ca molto ampi e in un numero d'iterazioni relativamente contenuto. Uno dei motivi per cui questo può verificarsi risiede nel fatto che gli AG campionano inizialmente, in maniera più o meno fine a seconda del numero d'individui utilizzato, lo spazio delle soluzioni del problema indirizzando la ricerca, di generazione in generazione, verso le zone più promettenti, cioè verso quelle zone caratterizzate da fitness media alta. Quanto velocemente questo avvenga dipende principalmente dal particolare schema di selezione adottato. Gli schemi caratterizzati da pressione selettiva bassa, come la selezione a torneo, hanno velocità di convergenza bassa. Questo permette di prolungare il campionamento dello spazio delle soluzioni per un certo numero di generazioni, che può essere anche grande e che dipende dal numero d'individui utilizzato, prima che gli sforzi dell'algoritmo genetico si concentrino nella zona a fitness più alta. Gli schemi caratterizzati da pressione selettiva alta tendono invece a far convergere gli individui nella zona più promettente con velocità elevata, addirittura esponenziale nel caso della selezione proporzionale. L'algoritmo genetico può comunque converge, anche se con probabilità più basse quando si utilizzano metodi di selezione con pressione selettiva bassa, verso un ottimo locale, per esempio quando il campionamento iniziale ne riconosce un intorno come la zona più promettente. In tal caso, se il paesaggio della fitness non è sufficientemente regolare, l'algoritmo genetico può rimanere intrappolato nell'ottimo locale e non riuscire ad allontanarsene neanche grazie al crossover e alla mutazione. Quando questo avviene si parla di convergenza prematura o convergenza verso un ottimo locale. Si consideri il caso in cui sia possibile definire due differenti funzioni di fitness per uno stesso problema. La figura 2.12 illustra due possibili paesaggi d'idoneità per la ricerca del massimo nell'intervallo $[0, \pi]$; nel caso (a) la funzione di fitness è definita come $f(x) = \frac{x}{32} + |\cos(16x)|$, mentre nel caso (b) è definita come $f(x) = \frac{x}{2} + |\cos(2x)|$. Nel caso (a) il paesaggio d'idoneità presenta molti ottimi locali e se l'algoritmo genetico indirizza la ricerca verso uno di essi diventa difficile allontanarsene poichè gli altri ottimi, compreso l'ottimo globale che si ha in corrispondenza del punto $x = \pi$, hanno valori di fitness molto vicini. Nel caso (b), invece, il compito è decisamente piu semplice, sia perchè il paesaggio d'idoneità presenta un numero minore di massimi locali, sia perchè i valori di fitness corrispondenti non sono vicini ed esistono molti altri punti dello spazio delle soluzioni con fitness maggiore facilmente raggiungibili tramite crossover e mutazione. Risulta dunque evidente che quando è possibile definire per uno stesso problema d'ottimizzazione più funzioni di fitness è opportuno sceglire quella che regolarizzi il più possibile il paesaggio d'idoneita o, equivalentemente, se ci si incorre in problemi di convergenza prematura è opportuno, quando possibile, ridefinire il compito dell'algoritmo genetico in modo da

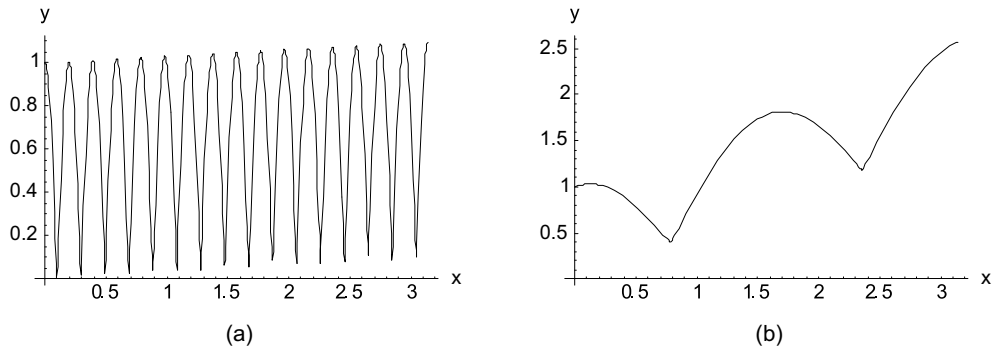


Figura 2.12: Rappresentazione grafica di due paesaggi d'idoneità per il problema della ricerca del massimo nell'intervallo $[0, \pi]$; nel caso (a) la funzione di fitness è definita come $f(x) = \frac{x}{32} + |\cos(16x)|$, mentre nel caso (b) è definita come $f(x) = \frac{x}{2} + |\cos(2x)|$.

regolarizzare il più possibile il paesaggio d'idoneità.

Gli AG possono essere dunque utilizzati con buoni risultati in tutti quei contesti in cui sia possibile adottare una codifica significativa delle soluzioni candidate, eventualmente adattando a essa gli operatori genetici, e definire una buona funzione di fitness per la loro valutazione proponendosi come uno strumento generale per la risoluzione di problemi complessi altrimenti difficilmente trattabili.

Capitolo 3

Automi Cellulari

3.1 Introduzione

Gli Automi Cellulari (AC) sono modelli di calcolo parallelo la cui evoluzione è regolata da leggi puramente locali. Nella sua definizione essenziale, un AC può essere descritto come uno spazio suddiviso in celle regolari, ognuna delle quali può trovarsi in un numero finito di stati. Ogni cella dell'AC ingloba, infatti, un automa finito, uno dei modelli di calcolo più semplici e noti in Informatica. Al tempo $t = 0$ le celle sono in uno stato arbitrario e l'AC evolve cambiando gli stati delle celle a passi discreti di tempo applicando simultaneamente a ognuna la stessa legge, o funzione, di transizione. L'input per ciascuna cella è dato dagli stati delle celle vicine e le condizioni di vicinato sono determinate da una relazione geometrica, invariante nel tempo e nello spazio. Si noti che l'AC, in alternativa, può essere visto come un reticolo regolare i cui nodi contengono l'automa finito.

A discapito della loro semplice definizione, gli AC possono dar luogo a comportamenti estremamente complessi [188]. A livello microscopico, infatti, le leggi che regolano la dinamica del sistema sono perfettamente note, ma questo non significa che da esse si possa dedurre in ogni caso il comportamento del sistema a livello macroscopico [30, 168]. In altri termini, la dinamica del sistema emerge in maniera non banale dalla mutua interazione delle sue componenti elementari e, anche a fronte di leggi d'interazione semplici, il comportamento macroscopico può risultare estremamente complesso.

L'equivalenza computazionale con la Macchina di Turing [35, 171] collocano gli AC tra i modelli di calcolo universali. Questo rende teoricamente possibile risolvere con gli AC qualsiasi problema "computabile" dato che tutto ciò che è computabile

è computabile tramite una Macchina di Turing (tesi di Church-Turing), quindi anche tramite un AC.

Nello specifico, gli AC si prestano particolarmente bene alla modellizzazione e simulazione di quei sistemi caratterizzati da numerosi costituenti elementari in mutua interazione. Un esempio particolarmante esplicativo è rappresentato dallo studio del comportamento dei fluidi (considerati a livello microscopico come sistemi di particelle) tramite modelli computazionali noti con il nome di Gas Reticolari [168]. Non meno importanti sono gli studi teorici che vedono gli AC come sistemi di calcolo parallelo [173, 46].

3.2 Breve storia degli Automi Cellulari

Nel '47 John von Neumann, matematico americano di origine ungherese, aveva intrapreso lo studio su quali fossero le caratteristiche e la complessità di un sistema che lo rendano capace di autoriproduzione. von Neumann morì prematuramente nel '57 e non ebbe tempo di completare il suo *Theory of self reproducing automata*, uscito comunque postumo nel '66 a cura di A. W. Burks [179]. La strada inizialmente intrapresa fu quella di un modello continuo, basato su un sistema di equazioni differenziali, per descrivere uno spazio in cui fluttuavano liberamente una sorta di robot assemblatore e innumerevoli copie dei pezzi di cui egli stesso era composto; il robot era programmato ad "agganciare" i pezzi nello spazio e ad assemblarli opportunamente per costruire una copia di se stesso. Le difficoltà a gestire un tale complesso modello portarono von Neumann nel '51, su suggerimento di Stanislaw Ulam, a cambiare radicalmente approccio: una scacchiera infinita, le cui celle quadrate inglobavano un automa finito (detto poi automa elementare), sostituì lo spazio tridimensionale continuo con uno bidimensionale discreto; il tempo divenne anch'esso discreto nel senso che le celle cambiavano stato in maniera sincrona in un "passo di calcolo", in relazione al proprio stato e a quello delle celle "vicine", dove la "vicinanza" era definita da una relazione spaziale fissa nello spazio e nel tempo (una vicinanza a croce fu quella scelta da von Neumann). Lo stato della cella individuava lo stato funzionale di un pezzo dell'assemblatore oppure era "quiescente", cioè individuava una porzione di spazio inattiva che poteva assumere una funzionalità (passare dallo stato quiescente a uno diverso) solo se "stimolata" da una o più celle vicine non in stato quiescente; le funzioni di transizione degli automi finiti sono le stesse dappertutto nello spazio e non variano nel tempo. A

questo punto l'assemblatore diventava un modificatore degli stati quiescenti delle celle, da cui era circondato, fino a costruire una copia di se stesso.

Successivamente agli studi di von Neumann sull'autoriproduzione, Codd [35] e Thatcher [171] hanno studiato le proprietà computazionali degli AC, dimostrandone, tra l'altro, l'universalità computazionale.

Grande interesse intorno agli AC si ebbe negli anni '70 grazie al Gioco delle Vita (Life) di John Horton Conway [70]. Si tratta di un AC bidimensionale con celle quadrate e vicinato della cella costituito dalla cella stessa e dalle 8 celle che la circondano. Ogni cella può assumere solo due stati: lo stato morto (stato 0) e lo stato vivo (stato 1). La funzione di transizione è data dalle seguenti regole: 1) una cella nello stato vivo passa allo stato morto se è in contato con meno di due celle nello stato vivo (muore per isolamento), oppure se è in contatto con più di tre celle in stato vivo (muore per sovrappopolazione); 2) una cella nello stato morto passa allo stato vivo (nasce) se è in contatto con esattamente tre celle nello stato vivo. Applicando queste semplici regole la popolazione di celle vive evolve continuamente assumendo configurazioni imprevedibili: in qualche caso la popolazione si estingue, in altri raggiunge configurazioni stabili o oscillanti, in altri ancora dà vita a strutture, i glider, in grado di muoversi nello spazio cellulare e interagire, in modo anche molto complesso, con altri glider o con altre strutture stabili o oscillanti. L'emergenza di comportamenti tanto complessi e imprevedibili in un sistema estremamente semplice come Life misero in luce le enormi potenzialità degli AC.

Alcune tra le ricerche teoriche e applicative più interessanti sugli AC per gli argomenti discussi in questo lavoro sono sinteticamente descritti nel paragrafo 3.6.

3.3 Definizione informale di Automa Cellulare

Una prima definizione informale di Automa Cellulare, quale si può dedurre dal primo lavoro di von Neumann, può essere data elencandone le proprietà fondamentali.

Definizione 3.3.1 (Definizione informale di Automa Cellulare). Un Automa Cellulare è caratterizzato dalle seguenti proprietà fondamentali:

- è formato da uno spazio d-dimensionale suddiviso in celle regolari (triangoli, quadrati, esagoni, cubi, ecc.) o, equivalentemente, da un reticolo regolare d-dimensionale;
- il numero di stati della cella è finito;

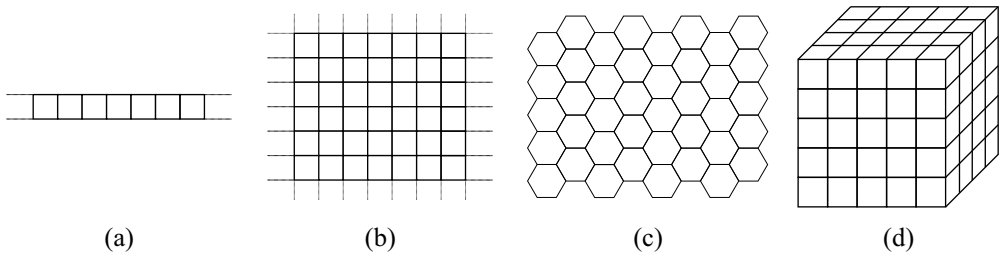


Figura 3.1: Esempi di spazi cellulari (a) unidimensionale, (b) bidimensionale con celle quadrate, (c) bidimensionali con celle esagonali e (d) tridimensionale con celle cubiche.

- l'evoluzione avviene a passi discreti;
- ogni cella cambia di stato simultaneamente a tutte le altre in accordo alla stessa regola di transizione;
- la regola di transizione dipende dallo stato della cella stessa e dallo stato delle celle vicine;
- la relazione di vicinanza è locale, uniforme e invariante nel tempo.

3.3.1 Dimensione e geometria dell'Automa Cellulare

La definizione di Automa Cellulare richiede, dunque, la discretizzazione dello spazio in celle. Per gli automi cellulari unidimensionali l'unica possibilità è una sequenza di celle allineate una a fianco all'altra, ovvero un reticolo unidimensionale. Per automi cellulari di dimensioni superiori esistono diverse alternative; per automi cellulari bidimensionali, per esempio, si possono adottare reticoli triangolari, quadrati o esagonali, mentre per automi cellulari tridimensionali si scelgono, solitamente, celle cubiche. La figura 3.1 illustra alcuni esempi di spazi cellulari in una, due e tre dimensioni.

Per quanto riguarda gli automi cellulari bidimensionali, sebbene la tassellazione quadrata sia facilmente rappresentabile attraverso una matrice e non presenti problemi nella rappresentazione grafica (per esempio ogni elemento della matrice può essere visualizzato utilizzando un pixel della matrice dello schermo), in alcune applicazioni può presentare problemi di anisotropia (l'argomento è ripreso più nei dettagli nel paragrafo 3.6.2). Quando questo si verifica si preferisce adottare una tassellazione esagonale che, per AC bidimensionali, è quella con anisotropia più

bassa [187], può rendere le simulazioni più realistiche e, in alcuni casi, è indispensabile per modellare correttamente alcuni fenomeni [181]. Purtroppo non esiste un equivalente tridimensionale dello spazio cellulare esagonale; come si vedrà (ancora nel paragrafo 3.6.2) questo può richiedere il ricorso a spazi di dimensione quattro.

3.3.2 Numero di stati della cella

Il numero di stati della cella deve essere finito ed è determinato in relazione al particolare contesto di studio o d'applicazione. Nei primi studi teorici successivi a von Neumann che hanno visto gli AC come modelli computazionali astratti [35, 171] il numero di stati della cella era, solitamente, abbastanza piccolo. Con soli due stati, per esempio, è possibile rappresentava, nella configurazione iniziale (specificazione dello stato, in questo caso 0 o 1, di tutte le celle dello spazio cellulare al tempo $t = 0$), l'informazione che l'AC deve elaborare.

Anche quando l'automa cellulare è utilizzato per descrivere sistemi di particelle e modellarne le interazioni, il numero di stati è abbastanza contenuto [168, 181].

Al contrario, quando si studiano sistemi che possono trovarsi in un continuum di possibili stati, può essere necessario un numero di stati della cella abbastanza grande perchè il modello sia significativo [59].

3.3.3 Relazione di vicinanza

La relazione di vicinanza della cella, cui ci si riferirà con il nome di cella centrale, dipende dalla geometria delle celle. Secondo la definizione data, deve godere delle seguenti proprietà: 1) deve essere locale, cioè deve coinvolgere solo un numero limitato di celle in prossimità della cella centrale; 2) deve essere omogenea, cioè la stessa per ogni cella dello spazio cellulare; 3) deve essere invariante nel tempo.

Per AC unidimensionali si è soliti riferirsi al vicinato in termini di raggio, r, che definisce un vicinato composto da $n = 2r + 1$ celle [186]. Per esempio, un raggio $r = 1$ identifica un vicinato di $n = 2r + 1 = 3$ celle: la cella centrale, la cella adiacente a sinistra e la cella adiacente a destra. La figura 3.2 illustra due esempi di vicinato con raggio $r = 1$ ed $r = 2$ per un automa cellulare unidimensionale.

Nel caso di AC bidimensionali con tassellazione quadrata i vicinati più utilizzati sono quello di von Neumann e quello di Moore. Il primo è composto dalla cella centrale e dalle celle a nord, est, ovest e sud, mentre il secondo contiene anche quelle di nord-ovest, nord-est, sud-ovest e sud-est. Un tipico vicinato per automi cellulari bidimensionali esagonali è, invece, composto dalle celle a nord, nord-est, sud-est,

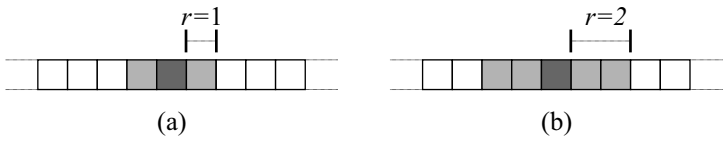


Figura 3.2: Esempio di vicinato con raggio (a) $r = 1$ e (b) $r = 2$ per un automa cellulare unidimensionale. Le celle in grigio scuro identificano la cella centrale, quelle in grigio chiaro le vicine.

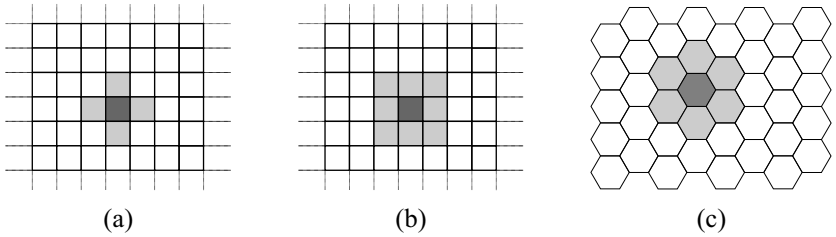


Figura 3.3: Vicinati di von Neumann (a) e di Moore (b) per un automa cellulare bidimensionale con tassellazione quadrata e vicinato esagonale (c) per un automa cellulare bidimensionale con tassellazione esagonale. Le celle in grigio scuro identificano la cella centrale, quelle in grigio chiaro le vicine.

sud, sud-ovest e nord-ovest. La figura 3.3 illustra (a) il vicinato di von Neumann e (b) quello di Moore per automi cellulari con tassellazione quadrata e (c) il tipico vicinato per automi cellulari con tassellazione esagonale. Ovviamente è possibile definire relazioni di vicinanza differenti da quelle illustrate. Nella simulazione della diffusione di gas in un ambiente, ad esempio, è possibile utilizzare la relazione di vicinanza di Margolus[1] [173].

3.3.4 Funzione di transizione di stato della cella

A ogni passo dell'AC, la funzione di transizione è applicata simultaneamente a tutte le celle dello spazio cellulare, determinando il nuovo stato di ognuna in funzione

[1]La relazione di vicinanza di Margolus non gode della proprietà d'invarianza temporale. Infatti il vicinato della cella cambia a seconda del passo, pari o dispari, dell'AC. Ai passi pari il vicinato è formato dalla cella centrale e dalle celle a nord, est, e nord-est; ai passi pari, invece, il vicinato è formato dalla cella centrale e dalle celle a sud, ovest, e sud-ovest. Si noti, tuttavia, che un AC che utilizzi tale relazione di vicinanza è perfettamente "legale". E' infatti possibile dimostrare che, dato un AC con relazione di vicinanza di Margolus, è possibile costruire un AC perfettamente equivalente che soddisfi tutte le proprietà richieste dalla definizione.

dello stato delle celle del vicinato. In tal modo la computazione dell'AC assume caratteristiche di parallelismo e di decentralizzazione.

Quando il numero di stati è piccolo, si è soliti definire la regola di transizione tramite una tabella (look-up table) che specifica il nuovo stato della cella centrale per ogni possibile configurazione del vicinato [188]. Al contrario, quando il numero di stati dell'AC è troppo grande, la funzione di transizione viene solitamente definita tramite l'esplicitazione di un algoritmo [59].

3.4 Definizione formale di Automa Cellulare

In letteratura è possibile incontrare numerose definizioni di AC, ognuna delle quali si adatta meglio delle altre ai particolari contesti d'applicazione. La definizione formale presentata in questo paragrafo coincide con la definizione di AC omogeneo deterministico. Questo, del resto, non è limitativo poichè la maggior parte delle altre definizioni possono essere a essa ricondotte.

Come accennato in precedenza, ogni cella dell'AC ingloba un identico automa finito. Nel caso specifico dell'AC omogeneo deterministico, si tratta di un automa finito anch'esso omogeneo e deterministico. E' opportuno, quindi, premetterne la definizione.

3.4.1 L'automa finito

L'automa finito (af), o automa a stati finiti, è probabilmente il modello di calcolo più semplice in informatica. Intuitivamente, un af è un sistema che può trovarsi in un numero finito di stati differenti e, come conseguenza di qualche ingresso, può effettuare una transizione da uno stato a un altro.

Come nel caso degli AC, in letteratura è possibile incontrare numerose definizioni di af. L'af presentato di seguito corrisponde alla definizione di af come riconoscitore (accettore) di linguaggi [71]. Il modello di af utilizzato negli AC, presentato subito dopo, è un modello ulteriormente semplificato che prende il nome di automa elementare.

Definizione 3.4.1 (Definizione formale di automa finito deterministico come riconoscitore di linguaggi). Un automa finito deterministico come riconoscitore di linguaggi è formalmente definito come una quintupla

$$af =< Q, I, \sigma, q_0, F >$$

dove:

Q è l'insieme finito degli stati dell'af;

I è l'insieme finito dei simboli d'ingresso (input);

$\sigma : I \times Q \to Q$ è la funzione di transizione che modifica gli stati in funzione dell'input;

$q_0 \in Q$ è lo stato iniziale dell'af;

$F \subset Q$ è l'insieme degli stati finali dell'af.

Definizione 3.4.2. Sia I^* l'insieme di tutte le possibili stringhe che si possono costruire sull'insieme dei simboli d'ingresso I. Il linguaggio L accettato dall'af è definito nel seguente modo:

$$L = \{x \in I^* | \sigma(q_0, x) \in F\}$$

dove $\sigma(q_0, x)$ indica lo stato in cui si trova l'af al termine della computazione. Pertanto una stringa $x \in I^*$ è accettata dall'af se questo, dopo aver letto tutta la stringa, si trova in uno stato finale.

Esempio 3.4.1. La figura 3.4 illustra un af che riconosce le stringhe binarie di lunghezza qualsiasi in cui compaiano almeno tre simboli 0 consecutivi. In questo caso $I = \{0, 1\}$, $Q = \{q_i, q_1, q_2, q_f\}$, $q_0 = q_i$ ed $F = \{q_f\}$. Al passo $t = 0$ l'af si trova nello stato iniziale q_i e viene letto il primo simbolo della stringa. La funzione di transizione σ è applicata tante volte quanti sono i simboli della stringa d'ingresso e determina il nuovo stato dell'af in base allo stato attuale e all'input. Esaurito l'input, la stringa è riconosciuta se e solo se l'af si trova nello stato finale q_f. La stringa $x_1 = 100110$ non è riconosciuta dall'af. Infatti la sequenza di transizioni di stato $\sigma(q_i, x_1)$ non termina nello stato finale q_f:

$$q_i \overset{\sigma(1,q_i)}{\longmapsto} q_i \overset{\sigma(0,q_i)}{\longmapsto} q_1 \overset{\sigma(0,q_1)}{\longmapsto} q_2 \overset{\sigma(1,q_2)}{\longmapsto} q_i \overset{\sigma(1,q_i)}{\longmapsto} q_i \overset{\sigma(0,q_i)}{\longmapsto} q_1$$

La stringa $x_2 = 100011$ è invece riconosciuta dall'af poichè la sequenza di transizioni di stato $\sigma(q_i, x_2)$ termina nello stato finale q_f:

$$q_i \overset{\sigma(1,q_i)}{\longmapsto} q_i \overset{\sigma(0,q_i)}{\longmapsto} q_1 \overset{\sigma(0,q_1)}{\longmapsto} q_2 \overset{\sigma(0,q_2)}{\longmapsto} q_f \overset{\sigma(1,q_f)}{\longmapsto} q_f \overset{\sigma(1,q_f)}{\longmapsto} q_f$$

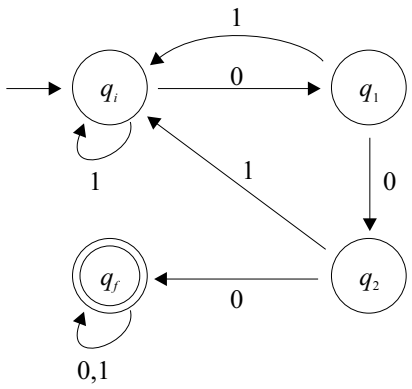


Figura 3.4: Automa finito che riconosce le stringhe binarie in cui compaiano almeno tre simboli 0 consecutivi. I nodi indicano gli stati dell'af, gli archi le transizioni. All'inizio della computazione, l'af si trova nello stato iniziale q_i; se viene letto il simbolo d'ingresso 0, allora effettua una transizione verso lo stato q_1; al contrario, se viene letto il simbolo 1, l'af rimane nello stato iniziale q_i. L'af riconosce la stringa in ingresso se alla fine della computazione si trova nello stato finale q_f.

Le celle dell'AC inglobano un tipo di af, detto elementare (ae), che è un'ulteriore semplificazione del semplice modello appena descritto: non è rilevante se l'ae termina il suo calcolo in uno stato finale o in qualsiasi altro stato. Ciò che importa è il cambiamento di stato in sè e la distinzione tra l'insieme degli stati Q e l'insieme degli stati finali $F \subset Q$ non è presente. L'insieme finito dei simboli d'ingresso è implicitamente definito dal numero di stati dell'ae e dal numero di celle del vicinato.

Definizione 3.4.3 (Definizione formale di automa elementare). L'automa elementare è formalmente definito come una tripla

$$a =< Q, I, \sigma >$$

dove:

Q è l'insieme finito degli stati dell'ae;

$I \equiv Q^n$ è l'insieme finito dei simboli d'ingresso, essendo n il numero di celle del vicinato;

$\sigma : I \rightarrow Q$ è la funzione di transizione che modifica gli stati in funzione dell'input.

A questo punto, dopo aver premesso le definizioni di automa finito e di automa elementare, si può definire formalmente l'AC omogeneo deterministico.

Definizione 3.4.4 (Definizione formale di automa cellulare). L'Automa Cellulare è formalmente definito come una quadrupla

$$A =< \mathbb{Z}^d, Q, X, \sigma >$$

dove:

$\mathbb{Z}^d = \{i \equiv (i_1, i_2, \ldots, i_d) \mid i_k \in \mathbb{Z}\ \forall k = 1, 2, ..., d\}$ è l'insieme dei punti del reticolo d-dimensionale che definisce lo spazio cellulare dell'AC; $\mathbb{Z}$ è l'insieme dei numeri interi;

Q è l'insieme finito degli stati dell'automa elementare;

$X = \{\xi_0, \xi_1, \ldots, \xi_{m-1}\}$ è l'insieme finito degli m vettori d-dimensionali

$$\xi_j = \{\xi_{j_1}, \xi_{j_2}, \ldots, \xi_{j_d}\}$$

che definiscono l'insieme

$$V(X, i) = \{i + \xi_0, i + \xi_1, \ldots, i + \xi_{m-1}\}$$

delle coordinate delle celle vicine alla generica cella i di coordinate $(i_1, i_2, \ldots, i_d)$. X è detto indice o relazione di vicinanza[2].

$\sigma : Q^m \to Q$ è la funzione di transizione locale dell'automa elementare.

Esempio 3.4.2. Si consideri un automa cellulare bidimensionale con vicinato di von Neumann (figura 3.3a) e la cella i di coordinate $(7, 7)$. La relazione di vicinanza che definisce il vicinato di von Neumann è

$$X = \{\xi_0, \xi_1, \xi_2, \xi_3, \xi_4\} = \{(0, 0), (0, -1), (1, 0), (0, 1), (-1, 0)\}$$

Pertanto, l'insieme delle coordinate delle celle vicine alla cella i sono definite dall'insieme

$$V(X, i) = \{i + \xi_0, i + \xi_1, i + \xi_2, i + \xi_3, i + \xi_4\} = \{(7, 7) + (0, 0),$$
$$(7, 7) + (0, -1), (7, 7) + (1, 0), (7, 7) + (0, 1), (7, 7) + (-1, 0)\} =$$
$$= \{(7, 7), (7, 6), (8, 7), (7, 8), (6, 7)\}$$

[2]Per il seguito imponiamo che ξ_0 sia il vettore nullo e, di conseguenza, che ogni cella appartenga al proprio vicinato. Ci si riferirà a tale cella con il nome di cella centrale. Tuttavia è senz'altro pensabile un AC in cui la cella non faccia parte del proprio vicinato.

Definizione 3.4.5. Sia $C = \{c|c : \mathbb{Z}^d \to Q\}$ l'insieme delle possibili assegnazioni di stato ad A (insieme delle configurazioni dell'AC) e $c(i)$ lo stato della cella i nella configurazione c. La funzione di transizione globale dell'AC è definita come segue:

$$\begin{aligned} \tau : C &\longrightarrow C \\ c &\mapsto \tau(c) \end{aligned}$$

essendo

$$\tau(c)(i) = \sigma(c(V(X,i))) = \sigma(c(i+\xi_0), c(i+\xi_1), \ldots, c(i+\xi_{m-1}))$$

Definizione 3.4.6. Si definisce stato quiescente uno stato $q_0 \in Q$ tale che

$$\sigma(q_0, q_0, \ldots, q_0) = q_0$$

.

3.5 Studi teorici sugli Automi Cellulari

Di seguito sono illustrate alcune interessanti ricerche teoriche sugli AC. Poichè la maggior parte di esse riguardano gli AC unidimensionali è utile, per una migliore comprensione degli argomenti trattati, premettere alcune definizioni e convenzioni ormai divenute di uso comune.

3.5.1 Automi Cellulari unidimensionali

Gli AC più semplici, almeno dal punto di vista della loro costruzione, sono gli AC elementari [186]. Si tratta di AC unidimensionali di N celle con $k = 2$ stati (0 e 1), raggio del vicinato $r = 1$ e condizioni periodiche al contorno (lo spazio cellulare unidimensionale è visto come un anello in cui la prima e l'ultima cella sono adiacenti). La figura 3.5 illustra un esempio di AC con condizioni periodiche al contorno. La scelta di adottare uno spazio cellulare così fatto nasce, in generale, dall'impossibilità di gestire spazi cellulari infiniti. Una soluzione ad anello, pur essendo finita, definisce comunque uno spazio illimitato in cui l'AC può evolveresi.

La regola di transizione σ della cella è espressa sotto forma di tabella (look-up table). Per esempio, se si indica con η una generica configurazione del vicinato (il numero di configurazioni del vicinato è dato da $k^{2r+1} = k^n$, nel caso degli AC

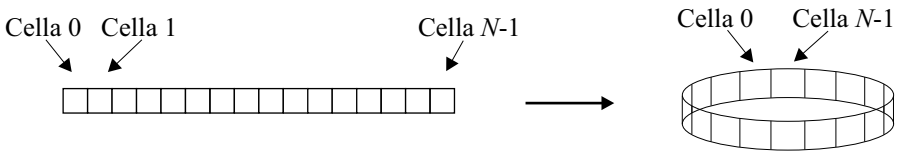


Figura 3.5: Esempio di automa cellulare unidimensionale con condizioni periodiche al contorno. La prima e l'ultima cella dello spazio cellulare risultano vicine nella rappresentazione ad anello.

elementari $2^3 = 8$), la seguente regola di transizione determina il nuovo stato, $s = \sigma(\eta)$, della cella centrale:

η	000	001	010	011	100	101	110	111
s	0	0	1	1	0	1	1	0

Si noti che le otto possibili configurazioni del vicinato sono elencate in ordine crescente secondo i valori binari che rappresentano. Infatti il numero binario 000 corrisponde al numero decimale 0, il numero binario 001 corrisponde al numero decimale 1, e così via; il numero binario 111 corrisponde, infine, al numero decimale 7. Adottando questa convenzione, una qualsiasi regola di transizione per AC elementari può essere definita semplicemente elencando i nuovi stati della cella centrale. La precedente regola può essere definita, pertanto, nel seguente modo:

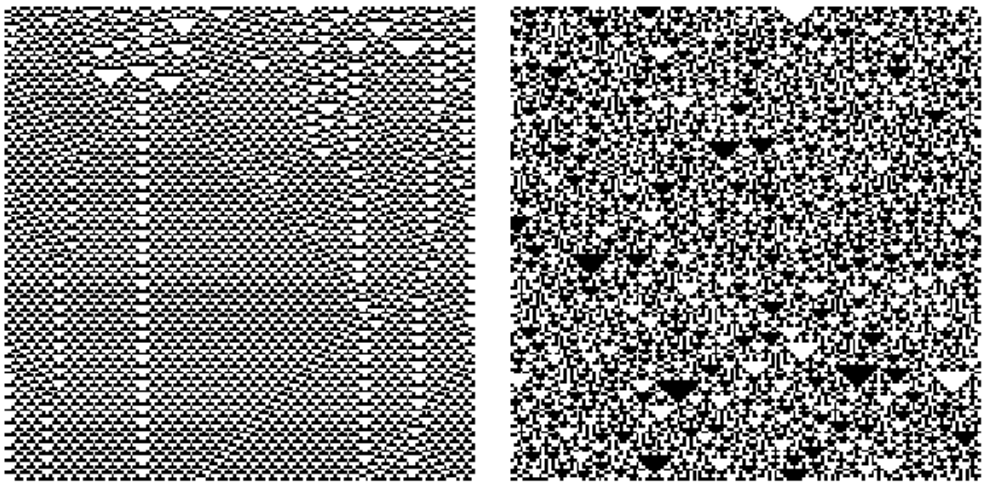
$$\sigma_{00010111} \equiv \begin{array}{cccccccc} 000 & 001 & 010 & 011 & 100 & 101 & 110 & 111 \\ 0 & 0 & 1 & 1 & 0 & 1 & 1 & 0 \end{array}$$

Anche in questo caso la regola di transizione σ definisce un numero binario. Pertanto si è soliti riferirsi alle regole di transizione degli AC elementari attraverso il numero decimale corrispondente al numero binario definito dalla regola. Per esempio:

$$\sigma_{00000000} \equiv \sigma_0$$
$$\ldots$$
$$\sigma_{00110110} \equiv \sigma_{54}$$
$$\ldots$$
$$\sigma_{11111111} \equiv \sigma_{255}$$

La figura 3.6 illustra due esempi di AC elementari.

Quando $r > 1$ il numero di configurazioni del vicinato cresce rapidamente. Per esempio, per $(k, r) = (2, 2)$ allora $k^{2r+1} = k^n = 2^5 = 32$. Di conseguenza, il numero complessivo di regole di transizione diviene 2^{32}=4294967296, rendendone



(a) (b)

Figura 3.6: I primi 100 passi di calcolo degli AC elementari (a) σ_{54} e (b) σ_{150}. La configurazione iniziale (prima riga dei due grafici) è generata in maniera casuale in modo che ogni cella possa assumere stato 0 (colore bianco) o 1 (colore nero) con eguale probabilità. L'evoluzione temporale è visualizzata dall'alto verso il basso come sequenza di righe.

di fatto impossibile un'analisi esaustiva. Lo stesso vale se si considerano AC con $(k, r) = (3, 1)$: in tal caso il numero di possibili regole di transizione è 3^{27}. In questi casi si suole restringere il campo d'indagine a particolari sottoclassi, come gli AC legali totalistici [186].

Un AC è considerato legale se preserva lo stato quiescente (solitamente lo stato nullo), cioè se lo stato della cella centrale di un vicinato in cui tutte le celle siano nello stato quiescente al passo t rimane nello stato quiescente al passo $t + 1$. Gli AC totalistici sono, invece, quegli AC caratterizzati dal fatto che il nuovo stato della cella dipende esclusivamente dalla somma dei valori degli stati delle celle del vicinato al passo precedente. In tal modo, per esempio, le regole legali totalistiche per AC con k stati e $r = 1$ passano da k^{k^3} a $k^{[k^2(1+k)-1]/2}$. Quindi, considerando $k = 2$ si passa da 256 a 32 regole legali, mentre per k=3 si passa da 3^{27} a 3^{17} regole legali totalistiche. Tuttavia, anche il numero di regole legali totalistiche cresce molto rapidamente con k ed r e per questo lo studio delle poroprietà degli AC si è concentrata per lo più su AC con pochi stati e vicinati di poche celle.

Come per gli AC elementari, anche gli AC totalistici possono essere specificati tramite un numero, c, detto codice della regola. Il codice c è definito in modo tale

che il coefficiente di 2^i nella sua rappresentazione binaria sia il valore atteso per lo stato della cella centrale quando la somma degli stati delle celle del vicinato sia pari a i. Per esempio, per $k = 2$ ed $r = 2$ il codice

$$c = 20 = 0 \cdot 2^5 + 1 \cdot 2^4 + 0 \cdot 2^3 + 1 \cdot 2^2 + 0 \cdot 2^1 + 0 \cdot 2^0$$

definisce la seguente regola totalistica $\sigma_{c=20}$ sotto forma di look-up table:

$$\sigma_{c=20} \equiv \begin{array}{cccccccc} \Sigma & 5 & 4 & 3 & 2 & 1 & 0 \\ s & 0 & 1 & 0 & 1 & 0 & 0 \end{array}$$

essendo Σ la somma degli stati delle celle del vicinato ed s il nuovo stato della cella centrale.

3.5.2 Classi di complessità e computazione universale

Wolfram [186, 188] ha proposto una classificazione degli AC unidimensionali con pochi stati e vicinati di poche celle in base al loro comportamento qualitativo, individuando quattro classi di complessità:

classe 1 gli AC della classe 1, indipendentemente dallo stato iniziale, convergono ad uno stato finale uniforme;

classe 2 gli AC della classe 2 convergono verso stati finali in cui semplici strutture rimangono sempre uguali o si ripetono nell'arco di pochi passi di calcolo;

classe 3 gli AC della classe 3 sono caratterizzati da un comportamento estremamente caotico;

classe 4 gli AC della classe 4 presentano sia comportamenti ordinati sia caotici. In questa classe si osservano semplici strutture localizzate che, tuttavia, possono interagire tra loro in maniera estremamente complessa.

Benchè altri schemi di classificazione siano stati successivamente proposti, per esempio da Chatè e Manneville [27], da Gutowitz [76] e da McIntosh [124], quello di Wofram è sicuramente il più noto. Esempi di AC appartenenti alle quattro classi di complessità di Wolfram sono illustrati in figura 3.7.

Tra le classi di complessità, la classe 4 è risultata particolarmente interessante per la presenza di strutture (glider) in grado di propagarsi nello spazio e nel tempo, tanto da aver spinto Wolfram ad avanzare l'ipotesi che gli AC di tale classe possano essere capaci di computazione universale.

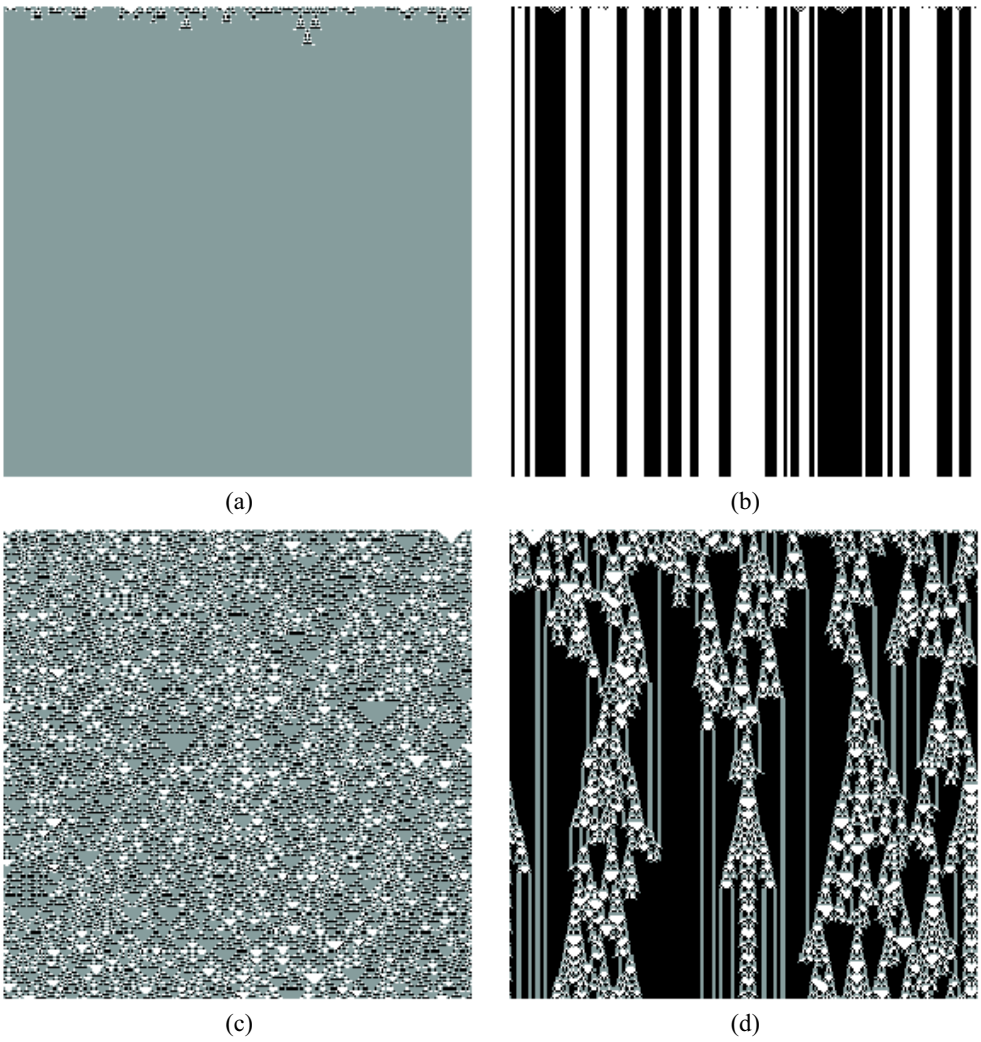


Figura 3.7: I primi 250 passi di calcolo degli AC legali totalistici $k = 3$, $r = 1$ (a) $\sigma_{c=1014}$, (b) $\sigma_{c=1008}$, (c) $\sigma_{c=1020}$ e (d) $\sigma_{c=2043}$. Gli AC illustrati appartengono rispettivamente alle classi di complessità 1, 2, 3 e 4. La configurazione iniziale, composta da 250 celle, è generata in maniera casuale in modo che ogni cella possa assumere stato 0 (colore bianco), 1 (colore grigio) o 2 (colore nero) con eguale probabilità. L'evoluzione temporale è visualizzata dall'alto verso il basso come sequenza di righe.

Un calcolatore può essere visto come un sistema in grado di trasformare una sequenza iniziale di bit in una sequenza finale codificante il risultato della computazione. La sequenza iniziale può essere considerata come la codifica dell'informazione che deve essere elaborata e del programma che deve essere eseguito, mentre la sequenza finale, o parte di essa, può essere considerata come risultato della computazione. Così, gli AC studiati da Wolfram possono essere visti come calcolatori: le configurazioni iniziali codificano dati e programma; le configurazioni finali (ottenute tramite un numero sufficiente di passi di calcolo) codificano il risultato della computazione. Se gli AC analizzati da Wolfram fossero capaci di computazione universale, dunque, con una prefissata regola di transizione dovrebbe essere possibile rappresentare nella configurazione iniziale ogni possibile programma.

In effetti, l'universalità computazionale degli AC è nota sin dalla loro nascita. La costruzione di von Neumann di una macchina in grado di autoriprodursi si basa, infatti, sulla dimostrazione dell'esistenza di un calcolatore/costruttore universale in un AC con 29 stati [179]. In seguito, Codd [35], Thatcher [171], Berlekamp, Conway e Guy [20] e Fredkin e Toffoli [66], solo per citarne alcuni, hanno dimostrato l'universalità computazionale per più semplici AC.

L'ipotesi di universalità computazionale per gli ancora più semplici AC studiati da Wolfram nasce dall'osservazione che i glider possano operare come "elaboratori" dell'informazione codificata nella configurazione iniziale. Per mezzo dei glider lo stato di una cella in una particolare posizione dello spazio cellulare può, infatti, influenzare nel tempo gli stati di celle in posizioni arbitrariamente lontane (figura 3.7d). Inoltre i glider possono interagire tra loro in maniera estremamente complessa e tramite essi è teoricamente possibile riprodurre, così come dimostrato per il Gioco della Vita di John Horton Conway [20, 70], le porte logiche fondamentali di un computer digitale universale. E' noto, dunque, che alcuni AC siano capaci di computazione universale, ma rimane questione aperta quale sia quel particolare AC capace di computazione universale con il più piccolo numero di stati e di vicini possibile.

La possibilità di computazione universale implica che ogni computazione può essere eseguita tramite AC. Inoltre, i meccanismi di processazione dell'informazione osservati in molti processi naturali appaiono più vicini agli AC che alle Macchine di Turing. Per questo motivo tali processi possono essere simulati più efficientemente tramite AC che non tramite altri modelli di calcolo [188]. Se si pensa, infine, che con le conoscenze tecnologiche attuali un semplice AC unidimensionale universale con svariati milioni di celle può essere facilmente realizzato su un singolo wafer di silicio con tempo di clock bassissimo, dell'ordine dei miliardesimi di secondo, si capisce

che la questione relativa alla progettazione di hardware dedicato potrebbe aprire nuove prospettive in termini di performance per la simulazione di quei fenomeni (fisici, biologici, economici, ecc.) che ben si prestano a essere descritti in termini di automi cellulari [186].

3.5.3 Il margine del caos

L'ipotesi di Wolfram secondo cui semplici AC unidimensionali siano capaci di computazione universale è stata ripresa successivamente da Chris Langton. In un lavoro del 1990 dal titolo "Computation at the edge of caos: phase transition and emergent computation" [111], Langton ha dimostrato che un'opportuna parametrizzazione dello spazio delle regole permette di individuare con una certa approssimazione sia le relazioni tra le classi di complessità, sia le zone di tale spazio in cui si collocano gli AC delle varie classi.

Per parametrizzare lo spazio delle regole, Langton ha utilizzato il parametro λ [110] che misura la percentuale di transizioni non quiescenti nella funzione di transizione dell'AC. Il parametro λ è definito nel seguente modo:

$$\lambda = \frac{k^n - n_q}{k^n} = 1 - \frac{n_q}{k^n}$$

essendo k il numero di stati della cella, $n = 2r + 1$ il numero di celle del vicinato ed n_q il numero di transizione che terminano nello stato quiescente. Se $n_q = k^n$, tutte le transizioni della look-up table sono verso lo stato quiescente e $\lambda = 0$; se $n_q = 0$, invece, non esistono transizioni verso lo stato quiescente e $\lambda = 1$; infine, quando nella look-up table tutti gli stati sono rappresentati nella stessa misura $\lambda = 1 - 1/k$.

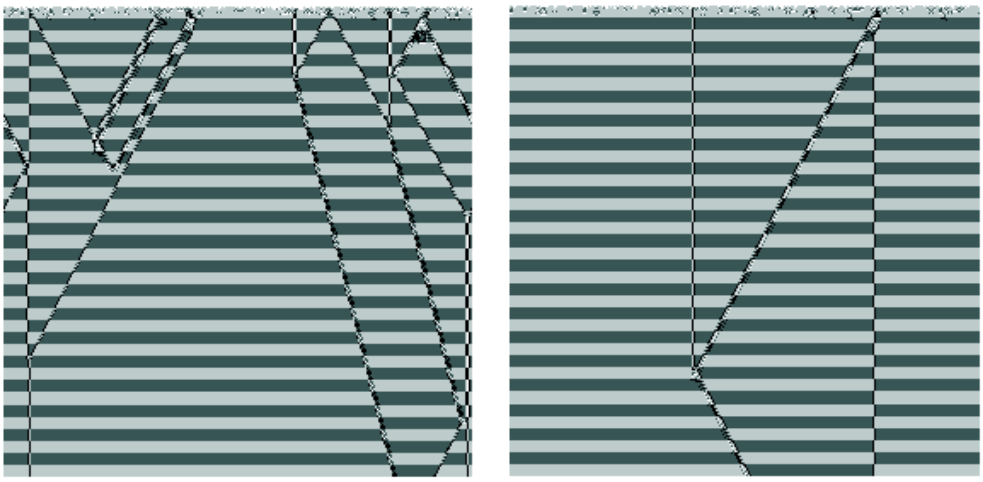
Esempio 3.5.1. Si consideri un AC con $k = 2$ stati in cui la regola di transizione mappi la metà dei 2^n vicinati nello stato 0 e l'altra metà nello stato 1. Di conseguenza $n_q = 2^n/2$, quindi

$$\lambda = 1 - \frac{n_q}{k^n} = 1 - \frac{\frac{1}{2}2^n}{2^n} = 1 - \frac{1}{2} = 0.5$$

Se, invece, $k = 4$ e la regola di transizione mappa esattamente un quarto dei 4^n vicinati in ognuno degli stati 0, 1, 2 e 3, allora $n = \frac{1}{4}4^n$ e

$$\lambda = 1 - \frac{n_q}{k^n} = 1 - \frac{\frac{1}{4}4^n}{4^n} = 1 - \frac{1}{4} = 0.75$$

I valori $\lambda = 0$ e $\lambda = 1 - 1/k$ rappresentano rispettivamente le regole di transizione più omogenee e più eterogenee che possono essere definite utilizzando k stati.



(a) (b)

Figura 3.8: Esempio di AC al "margine del caos". Le figure (a) e (b) mostrano l'evoluzione dello stesso AC con $k = 4$ stati ed $r = 1$ a partire da due differenti configurazioni iniziali. Le tonalità di grigio rappresentano i quattro possibili stati della cella, dal bianco per lo stato 0, al nero per lo stato 3.

Langton ha analizzato il comportamento di numerosi AC legali totalistici con $k = 4$ stati ed $r = 1$ al variare di λ nell'intervallo $[0, 0.75]$. I risultati hanno mostrato che per valori piccoli di λ il comportamento degli AC analizzati è essenzialmente ordinato, tipico delle classi di complessità 1 e 2, mentre per valori grandi di λ il comportamento osservato è essenzialmente caotico, tipico della classe 3. Tra queste due zone (ordine e caos), tuttavia, Langton ne ha osservata una terza molto piccola in prossimità del valore $\lambda = 0.45$. In questa zona, definita dallo stesso Langton "margine del caos", la dinamica dell'AC è in grado di generare sia strutture statiche, sia strutture in grado di propagarsi nello spazio e nel tempo, tipiche della classe 4 di Wolfram. La figura 3.8 illustra un esempio di un AC al "margine del caos". A differenza della prime due zone, solo al margine del caos l'informazione codificata nella configurazione iniziale dell'AC può propagarsi su lunghe distanze, condizione necessaria perchè si possa parlare di computazione. Inoltre, come si può anche notare in figura 3.8, nei pressi del margine del caos la dinamica degli AC dipende sensibilmente dalla configurazione iniziale. In tal modo, in accordo con l'ipotesi di Wolfram, l'AC è in grado di "discriminare" tra differenti informazioni (programma e dati) codificate nella stringa iniziale, rendendo così potenzialmente possibile un'effettiva elaborazione.

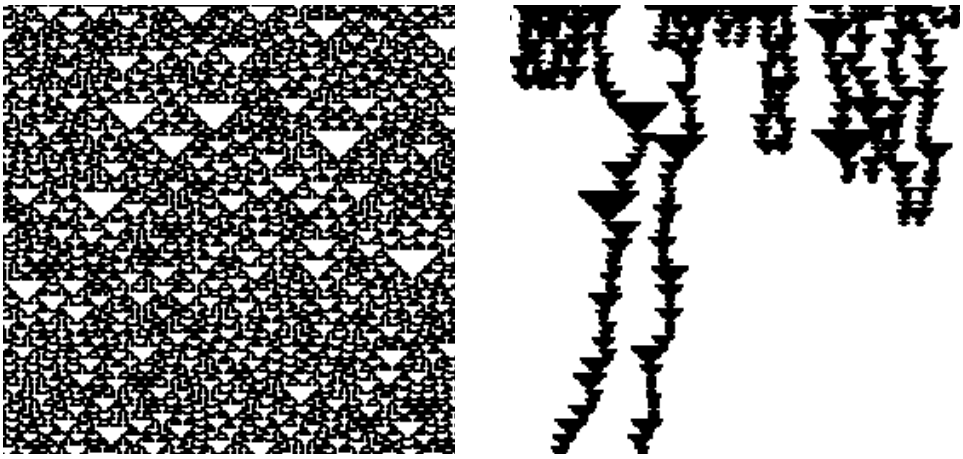
3.5.4 Meccanica computazionale

Lo studio sulla classificazione degli AC di Wolfram si colloca nell'ambito di un'analisi di tipo puramente macroscopico poichè si fonda esclusivamente sull'osservazione delle configurazioni spazio-temporali generate dagli AC, disinteressandosi completamente delle cause per cui tali configurazioni emergono. Diametralmente all'opposto, il tentativo di Langton di caratterizzare lo spazio delle regole attraverso il parametro λ si colloca all'interno di un approccio prettamente microscopico poichè si fonda esclusivamente sull'analisi delle interazioni locali definite dalla regola di transizione.

Entrambi gli approcci possono essere soggetti a critiche. Per esempio, lo schema di classificazione di Wolfram è soggettivo e, di conseguenza, differenti osservatori potrebbero non concordare sulla classe di appartenenza di particolari AC. Il lavoro di Langton, invece, si pone in qualche modo in antitesi rispetto al noto principio secondo cui in sistemi non lineari l'equazione che definisce il sistema (la regola di transizione nel caso degli AC) non determina direttamente il comportamento del sistema a lungo termine [46]. E' nota, infatti, la non assoluta precisione di λ: non tutte le regole al margine del caos risultano effettivamente complesse e, viceversa, regole complesse possono collocarsi anche al difuori di tale zona. In considerazione di questo, Wuensche [191, 189, 190] ha proposto il parametro Z che meglio sembra descrivere il comportamento degli AC a partire dall'analisi della regola di transizione.

La meccanica computazionale, proposta da Crutchfield [43] e applicata agli AC dallo stesso Crutchfield e da Hanson [79, 44, 77, 78], è un'approccio che si colloca a un livello intermedio tra il microscopico e il macroscopico nello studio della computazione emergente nei sistemi dinamici discreti. Applicata agli AC, l'obbiettivo della meccanica computazionale è scoprire un'appropriata configurazione base a partire dalla quale descrivere le componenti strutturali emergenti fondamentali e le loro interazioni.

Una configurazione base consiste in un insieme di linguaggi formali $\mathbf{\Lambda} = \{\Lambda^i,\ i = 0, 1, \ldots\}$. Per esempio, la configurazione base dell'AC elementare σ_{18}, illustrato in figura 3.9a, è $\mathbf{\Lambda} = \{\Lambda^0 = (0*)^+\}$, dove, nel contesto specifico della Meccanica Computazionale, il simbolo$*$ è un simbolo jolly che può assumere sia il valere 0 che il valore 1, mentre l'apice + indica una sequenza pari di ciò che è contenuto tra parentesi. In altri termini, in molte regioni nelle configurazioni spazio-temporali dell'AC σ_{18} gli stati delle celle sono istanze del linguaggio Λ^0, per esempio $0101\ldots01$, oppure $0000\ldots00$.



(a) (b)

Figura 3.9: Diagramma spazio-temporale (a) dell'AC elementare σ_{18} e (b) dello stesso AC in cui sono state "filtrate" le istanze del linguaggio Λ^0 ed evidenziate in nero le particelle emergenti.

Cruthfield e Hanson definiscono dominio regolare Λ^j una regione della configurazione spazio-temporale dell'AC invariante sia rispetto allo spazio che al tempo tale che sia un linguaggio regolare. Individuare un dominio regolare[3] significa, almeno in parte, comprendere la dinamica dell'AC. I domini regolari definiscono, infatti, motivi di "regolarità" nelle corrispondenti zone dei diagrammi spazio-temporali.

I domini possono essere filtrati mettendo in evidenza le zone di demarcazione. Tali zone sono, di fatto, glider che, nella terminologia tipica della meccanica computazionale, prendono il nome di particelle. Le particelle che emergono filtrando l'unico dominio regolare Λ^0 dell'AC σ_{18} sono istanze del linguaggio $\mathbf{P} = \{1(00)^n1,\ n = 0, 1, 2, \ldots\}$. La figura 3.9b illustra la configurazione spazio-temporale dell'AC σ_{18} in cui le celle di Λ^0 sono colorate in bianco, mentre le celle di $\mathbf{P}$ sono colorate in nero.

Per l'analisi del comportamento dell'AC, domini e particelle sono riportate in un catalogo. Nel caso dell'AC σ_{18} il catalogo è particolarmente semplice (tabella 3.1).

[3]I domini regolari sono spesso individuabili tramite l'osservazione diretta delle configurazioni spazio-temporali degli AC. Per i casi in cui questo risulti difficile, come per l'AC σ_{18}, Crutchfield e Hanson [43, 77] hanno sviluppato un metodo automatico per il riconoscimento dei domini regolari, chiamato $\epsilon - machine$.

Domini Regolari
$\Lambda^0 = \{(0*)^+\}$ $(* = 0\, oppure * = 1)$
Particelle
$\alpha \sim \mathbf{P} = \{1(00)^n 1,\ n = 0, 1, 2, \ldots\}$
Interazioni
$\alpha + \alpha \rightarrow \emptyset$ (Λ^0)

Tabella 3.1: Catalogo dell'AC elementare σ_{18}. Sono riportati i domini regolari, le paricelle emergenti e le loro interazioni.

Il risultato è che è che l'AC σ_{18} può essere analizzato in maniera strutturale, piuttosto che semplicemente definirlo caotico secondo la classificazione di Wolfram. Le particelle emergenti portano con sè vari tipi d'informazione relativi alle regioni della configurazione iniziale dell'AC da cui provengono. Le loro interazioni, invece, fungono da elaboratori d'informazione.

3.5.5 Algoritmi Genetici e computazione emergente negli Automi Cellulari

Crutchfield e Mitchell [45, 46] hanno applicato gli AG alla ricerca di AC unidimensionali in grado di eseguire compiti complessi che coinvolgano la coordinazione globale tra celle arbitrariamente lontane.

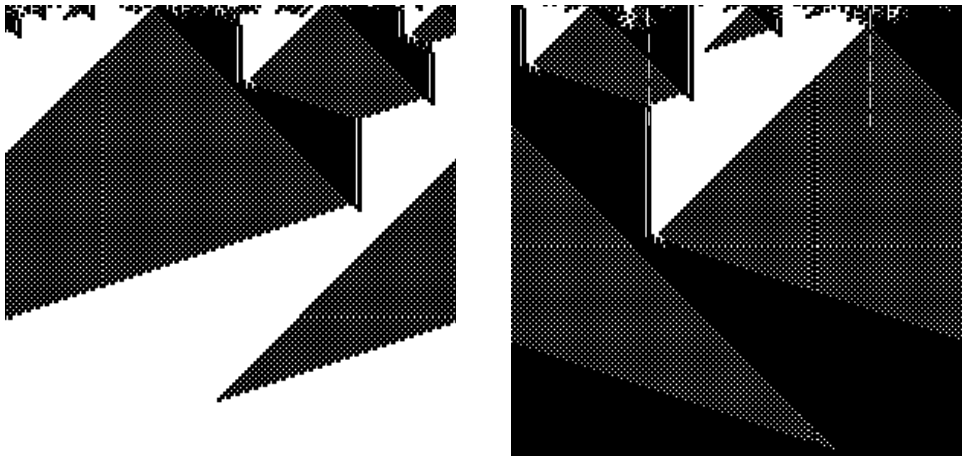
Un tipico esempio è costituito dal problema $\rho_0 > \rho_c$ che consiste nel valutare se nella configurazione iniziale dell'AC la frazione ρ_0 di celle nello stato 1 è maggiore di un valore prefissato ρ_c. Solitamente $\rho_c = 1/2$ e il problema si riduce a determinare se nella configurazione iniziale ci siano più celle nello stato 1 o nello stato 0. Nel primo caso l'obbiettivo per l'AC è convergere, in un prefissato numero di passi T_{max}, nella configurazione finale in cui le celle siano tutte nello stato 1, nel secondo caso nella configurazione finale in cui le celle siano tutte nello stato 0. Ovviamente il compito è indefinito nel caso in cui sia $\rho_0 = \rho_c$. La performance $P_N^I(\sigma)$ di un AC definito dalla regola di transizione σ è calcolata considerando I configurazioni iniziali di N celle generate casualmete, iterando l'AC per al più T_{max} passi e determinando la frazione di volte in cui la configurazione iniziale è classificata correttamente. Il compito $\rho_c = 1/2$, banale per modelli computazionali come la Macchina di Turing, è tutt'altro che semplice per un AC poichè non esiste un controllo globale che possa contare il numero di celle in un dato stato, anzi ogni cella può interagire soltanto con poche altre celle vicine.

Per i 256 AC elementari, la massima performance è all'incirca $P_N^{10^4} = 0.5$ per $N \in \{149, 599, 999\}$. Per AC $(k, r) = (2, 2)$ la massima performance è all'incirca $P_N^{10^4} = 0.58$ per $N = 149$ e $P_N^{10^4} = 0.5$ per $N \in \{599, 999\}$. Per AC $(k, r) = (2, 3)$ le performance osservate sono decisamente superiori e alcune strategie di risoluzione del problema particolarmente interessanti.

L'AG utilizzato per la ricerca di AC $(k, r) = (2, 3)$ in grado di risolvere il compito $\rho_c = 1/2$ è di tipo binario. Il genotipo codifica la regola di transizione dell'AC (soluzione candidata del problema) considerando i $k^{2r+1} = 2^{2 \cdot 3+1} = 128$ bit di output dell look-up table. Lo spazio di ricerca per l'AG contiene, in tal modo, 2^{128} soluzioni candidate e risulta troppo ampio per una ricerca esaustiva. Una popolazione iniziale di M genotipi è selezionata in maniera casuale. La fitness $F_N^I(\sigma)$ di un genotipo che rappresenta l'AC definito dalla regola di transizione σ è calcolata considerando I configurazioni iniziali di N celle generate casualmente, iterando l'AC per al più T_{max} passi e determinando la frazione di volte in cui la configurazione iniziale viene classificata correttamente. In ognuna delle G generazioni dell'AG, (1) è generato un nuovo insieme I di configurazioni iniziali dell'AC, (2) è calcolato il valore della funzione di fitness $F_N^I(\sigma)$ per ogni genotipo della popolazione, (3) i genotipi vengono ordinati in base al valore di fitness, (4) un numero E dei migliori è copiato nella nuova popolazione senza modifiche (schema steady-state elitistico), (5) i rimanenti $M - E$ genotipi sono ottenuti tramite crossover a singolo punto (con probabilità p_c) e mutazione (con probabilità p_m) sugli E genotipi migliori; lo schema adottato è con rimpiazzamento, cioè un cromosoma elitario può essere scelto più volte per la ricombinazione. I valori dei parametri adottati negli esperimenti brevemente descritti sotto sono: $M = 100$, $I = 100$, $E = 20$, $N = 149$, $T_{max} = 2N$, $p_c = 1.0$, $p_m = 0.016$ e $G = 100$.

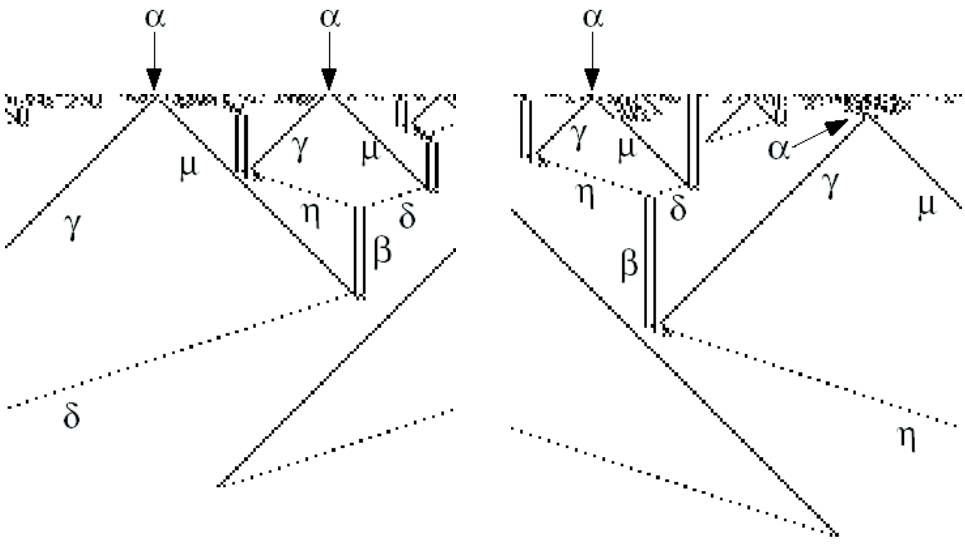
In ognuno dei 300 esperimenti eseguiti, l'AG ha evidenziato tre differenti strategie per la risoluzione del compito $\rho_c = 1/2$: default, espansione di blocchi, particellare. Tra queste la più interessante, e al contempo più performante, è risultata la strategia particellare. La figura 3.10 mostra un esempio di un AC, σ_{par}, che adotta la strategia particellare per risolvere il compito di classificazione $\rho_c = 1/2$. Le sue performace sono $P_{149}^{10^4}(\sigma_{par}) = 0.775$, $P_{599}^{10^4}(\sigma_{par}) = 0.740$ e $P_{999}^{10^4}(\sigma_{par}) = 0.728$.

Nella fase iniziale, la strategia dell'AC σ_{par} consiste nel mappare regioni ad alta densità di celle nello stato 1 in sottoconfigurazioni di celle tutte nello stato 1, e viceversa. Il bordo tra tali regioni e le regioni a scacchiera possono essere interpretate come segnali indicanti una situazione d'ambiguità. Le regioni con densità locale pari proprio a $\rho_c = 1/2$, dunque, non sono classificate direttamente, ma tale compito è demandato al comportamento e all'interazione di tali segnali.



(a) (b)

Figura 3.10: Diagrammi spazio-temporali di un AC, σ_{par}, evoluto tramite AG che risolve il problema di classificatione $\rho_c = 1/2$ adottando la strategia particellare: la figura (a) mostra l'evoluzione di una configurazione iniziale con $\rho_0 = 0.48$; la figura (b) mostra l'evoluzione di una configurazione iniziale con $\rho_0 = 0.51$.



(a) (b)

Figura 3.11: Diagrammi spazio-temporali filtrati dei domini regolari dell'AC σ_{par}, che risolve il problema di classificazione $\rho_c = 1/2$ adottando la strategia particellare: la figura (a) mostra l'evoluzione di una configurazione iniziale con $\rho_0 = 0.48$; la figura (b) mostra l'evoluzione di una configurazione iniziale con $\rho_0 = 0.51$.

Domini Regolari
$\Lambda^0 = \{0^+\}$, $\Lambda^1 = \{1^+\}$, $\Lambda^0 = \{(011)^+\}$
Particelle
$\alpha \sim \Lambda^1\Lambda^0$, $\beta \sim \Lambda^0\Lambda^1$, $\gamma \sim \Lambda^1\Lambda^2$, $\delta \sim \Lambda^2\Lambda^1$, $\eta \sim \Lambda^0\Lambda^2$, $\gamma \sim \Lambda^2\Lambda^0$
Interazioni
$\alpha \rightarrow \gamma + \mu$, $\beta + \gamma \rightarrow \eta$, $\mu + \beta \rightarrow \delta$, $\eta + \delta \rightarrow \beta$, $\eta + \mu \rightarrow \emptyset$, $\gamma + \delta \rightarrow \emptyset$

Tabella 3.2: Catalogo dell'AC elementare σ_{par}. Sono riportati i domini regolari, le paricelle emergenti e le loro interazioni.

Per comprendere con precisione la strategia dell'AC σ_{par}, può essere utile ricorrere agli strumenti propri della meccanica computazionale, descritta nel paragrafo precedente. La configurazione base di σ_{par}, visibile in figura 3.10, è formalmente definita come $\mathbf{\Lambda} = \{\Lambda^0 = 00^+, \Lambda^1 = 11^+, \Lambda^2 = (01)^+\}$, corrispondente alle regioni tutto bianco, tutto nero e a scacchiera. Filtrando il dominio regolare definito da $\mathbf{\Lambda}$ (figura 3.11) si notano differenti particelle, etichettate con lettere greche. La tabella 3.2 illustra il catalogo per l'AC σ_{par}. La dinamica dell'AC può essere ora interpretata in termini di particelle, che trasportano l'informazione sulle densità locali da cui hanno avuto origine. Le loro interazioni combinano e processano l'informazione: ad esempio si osserva che quando due particelle si scontrano sopravvive quella proveniente dalla zona della configurazione iniziale a densità più alta. La sequenza di urti particellari è dunque responsabile della classificazione della configurazione iniziale. La computazione, in definitiva, è effettuata a livello intermedio tra il microscopico e il pacroscopico, cioè al livello particellare.

3.5.6 Altri lavori teorici sugli Automi Cellulari

Gli studi illustrati in questo capitolo non esauriscono l'insieme delle ricerche teoriche sugli AC. Dopo la loro introduzione, infatti, numerosi contributi sono venuti da ricercatori da tutte le parti del mondo. Il problema della reversibilità negli AC è stato studiato, solo per citarne alcuni autori, da Moore [128], Myhill [133], Di Gregorio e Trautteur [61], Kari [100], e da Toffoli e Margolus [174]. Sulla scia dei lavori di Crutchfield, Hanson e Mitchell sulla meccanica computazionale e sull'evoluzione di AC in grado di eseguire compiti che richiedono coordinamento globale negli AC, Jiménez-Morales ha adottato un approccio evolutivo basato su AG per lo studio

di comportamenti collettivi non banali negli AC [97, 96, 98], mentre Tomassini e Vemzi [176] hanno studiato il problema di classificazione $\rho_c = 1/2$ per AC asincroni in cui l'aggiornamento dello stato delle celle non avviene simultaneamente. Altri interessanti lavori riprendono, sulla scia dell'originario studio di von Neumann, il problema dell'autoriproduzione negli AC. Tra essi si ricordano i lavori di Azpeitia e Ibáñez [14] e di Bilotta et al. [22]. Roli e Zambonelli [156] hanno studiato, infine, l'emergenza di strutture nella dinamica spazio temporale di AC dissipativi, cioè di AC visti come sistemi aperti in cui l'ambiente può influenzarne la dinamica.

3.6 Alcune applicazioni degli Automi Cellulari

In campo applicativo, gli AC si prestano particolarmente bene alla modellizzazione e simulazione di alcune classi di sistemi complessi, caratterizzati dell'interazione di numerosi componenti "elementari". L'ipotesi, largamente diffusa nella scienza tradizionale, secondo cui se il comportamento di un sistema è complesso il modello che lo descrive deve essere necessariamente di pari complessità, lascia il posto all'idea secondo cui il suo comportamento possa essere descritto, almeno in certi casi, in termini estremamente semplici, attraverso la specificazione delle leggi che definiscono le interazioni locali tra le componenti base del sistema [188].

In alcuni settori l'applicazione degli AC ha dato risultati paragonabili, se non superiori, a quelli ottenuti tramite approcci più tradizionali. Un esempio particolarmente significativo è l'applicazione alla modellizzazione del comportamento dei fluidi turbolenti attraverso modelli noti con il nome di Gas Reticolari e di modelli di Boltzmann su reticolo. Un altro importante campo d'applicazione è la "Vita Artificiale", disciplina che si occupa dello studio della vita e del comportamento nei sistemi naturali e artificiali. Negli ultimi anni, inoltre, gli AC sono stati applicati con successo nella modellizzazione di fenomeni naturali macroscopici. Di seguito sono brevemente descritti alcuni esempi applicativi in Vita Artificiale e nel settore dei Gas Reticolari. L'applicazione degli AC alla modellizzazione di fenomeni naturali macroscopici è, invece, discussa più approfonditamente nel capitolo successivo.

3.6.1 Automi Cellulari nella Vita Artificiale: il problema dell'autoriproduzione

La Vita Artificiale può essere definita come la disciplina che si occupa della vita e del comportamento di sistemi artificiali che "vivono" in ambienti artificiali. Chris Langton, uno dei suoi padri fondatori, definisce il nuovo settore di ricerca come lo studio di "sistemi *manufatti* che esibiscono comportamenti caratteristici dei sistemi viventi. La Vita Artificiale amplia la Biologia classica, che si occupa dell'analisi degli organismi viventi, tentando di *sintetizzare* comportamenti simili alla vita (like-life behaviors) nei computer o in altri media artificiali. ... la Vita Artificiale può contribuire in Biologia Teorica nell'inquadrare la *vita così come la conosciamo* nel più ampio contesto della *vita come potrebbe essere*" [112]. Di seguito è brevemente illustrato il filone di ricerca della Vita Artificiale, in cui gli AC hanno giocato un ruolo di primo piano, che si occupa del problema dell'autoriproduzione[4].

Fu lo stesso Langton a suggerire che gli AC potessero rappresentare un modello estremamente efficace per gli studi in Vita Artificiale [109]. In effetti, uno dei suoi precursori fu proprio John von Neumann che, già a partire dalla fine degli anni '40, si era dedicato allo studio dell'autoriproduzione degli organismi viventi adottando un approccio "artificiale" basato sugli AC.

Successivamente, prima Codd, alla fine degli anni '60, e poi Langton, intorno alla metà degli anni '80, hanno proposto modelli semplificati rispetto all'originario modello di autoriproduzione di von Neumann con strutture autoreplicanti. von Neumann era convinto che l'autoriproduzione dovesse essere definita in modo tale da inglobare la proprietà di computabilità universale; questo spiega la grande complessità del suo modello, che può essere visto essenzialmente a due differenti livelli: 1) l'AC vero e proprio, con 29 stati per la cella e una complicatissima funzione di transizione; 2) un "costruttore universale" (il robot assemblatore, ovvero una Macchina di Turing Universale) inglobato nello spazio cellulare.

Codd, pur condividendo l'ipotesi di von Neumann, riteneva che la complessità del suo modello fosse troppo elevata e ne propose uno alternativo con soli 8 stati [35]. Solo nel 1984, però, Langton propose un modello con strutture autoreplicanti (i Langton's Loop) non equivalenti a una Macchina di Turing dal punto di vista computazionale, dimostrando così che la proprietà di computabilità universale è sufficiente ma non necessaria all'autoriproduzione [108].

[4]Per un approfondimento più generale sulla Vita Artificiale e sulle sue applicazioni si vedano i proceedings delle conferenze *Artificial Life* che, a partire dalla prima tenutasi a Los Alamos nel 1987, si ripetono con cadenza biennale.

Bisogna, tuttavia, sottolineare una fondamentale differenza tra il punto di vista di Langton e quello di von Neumann sull'autoriproduzione. Come accennato in precedenza, von Neumann riteneva la proprietà di computabilità universale necessaria per ipotesi. Langton fece, tuttavia, notare come semplici organismi viventi, per i quali non sia stato possibile dimostrare la proprietà di computabilità universale, fossero capaci di autoriprodursi. Rinunciare all'universalità computazionale non implicava dunque, secondo Langton, perdere le caratteristica fondamentale del processo dell'autoriproduzione. Questo, tuttavia, significava accettare i casi di autoriproduzione banali in cui il fenomeno si origina non per le proprietà intrinseche dell'autoriproduttore ma per effetto della regola di transizione dell'AC (la regola di transizione di un AC elementare che assegna a ogni configurazione del vicinato la somma modulo 2 degli stati delle celle fornisce esempi di autoriproduttori banali). Per tale motivo Langton definì come fenomeni "autentici" di autoriproduzione solo quei fenomeni che dipendono principalmente dalle proprietà intrinseche dell'autoriproduttore e solo in minima parte, o per nulla, dalla "fisica" del sistema [108].

I modelli precedentemente descritti replicano particolari strutture (per esempio il Langton's Loop) definite nella configurazione iniziale dell'AC grazie a una funzione di transizione specificamente progettata in modo da operare efficacemente sulla particolare configurazione e non su altre. Chou e Reggia [33] hanno dimostrato, per la prima volta, che è possibile realizzare AC con funzioni di transizione più "generali" in grado di far emergere strutture autoreplicanti da configurazioni iniziali del tutto casuali. Tali strutture possono avere caratteristiche e forme differenti e interagire, in modalità anche molto complesse, con altre strutture che contestualmente emergono nello spazio cellulare.

Come già accennato, altri interessanti lavori che riprendono l'originario studio di von Neumann sul problema dell'autoriproduzione negli AC sono dovuti ad Azpeitia e Ibáñez [14] e a Bilotta et al. [22]. In particolare, quest'ultimo lavoro si colloca, in una certa misura, sulla linea tracciata da Chou e Reggia: strutture autoreplicanti emergono da configurazioni casuali e possono mutuamente interagire nello spazio cellulare. La novità metodologica sta nel fatto che la ricerca di tali strutture non è mirata, ma è un effetto diretto della ricerca di regole di transizione complesse dell'AC. La ricerca di tali regole è eseguita tramite Algoritmi Genetici, la cui funzione di fitness premia quelle regole che risultano maggiormente complesse secondo la definizione di complessità proposta da Wuensche [190]. Se da una parte è vero che la computabilità universale è solo condizione sufficiente ma non necessaria all'autoriproduzione [108], i risultati di questo studio sembrano indicare

che la complessità, così come proposta da Wuensche, invece lo sia. Successivi studi hanno, tuttavia, mostrato che anche regole non complesse possono far emergere strutture autoreplicanti nella dinamica dell'AC con la conseguenza che anche la condizione di complessità risulta non necessaria al fenomeno dell'autoriproduzione [147].

Come si evince dal filone di ricerca sull'autoriproduzione, dunque, la Vita Artificiale ha prodotto ipotesi di lavoro e risultati originali e di estremo interesse, sia dal punto di vista teorico che da quello delle possibili applicazioni. In questo contesto, gli AC hanno giocato, e continuano a giocare, un ruolo di primo piano.

3.6.2 Gas Reticolari e modelli di Boltzmann su reticolo

La dinamica dei fluidi è quella branca della fisica che si occupa del comportamento di gas e liquidi [107]. La fluidodinamica classica si fonda sulle equazioni di Navier-Stokes che formalizzano le leggi di conservazione della massa e dell'impulso:

$$\begin{aligned}\frac{\partial \rho u}{\partial t} &= -\frac{\partial \rho uu}{\partial x} - \frac{\partial \rho uv}{\partial y} - \frac{\partial p}{\partial x} + \mu\left(\frac{\partial^2 u}{\partial^2 x} + \frac{\partial^2 u}{\partial^2 y}\right)\\ \frac{\partial \rho v}{\partial t} &= -\frac{\partial \rho uv}{\partial x} - \frac{\partial \rho vv}{\partial y} - \frac{\partial p}{\partial y} + \mu\left(\frac{\partial^2 v}{\partial^2 x} + \frac{\partial^2 v}{\partial^2 y}\right)\end{aligned}$$

essendo u e v le velocità del fluido nelle direzioni x e y di un sistema di riferimento cartesiano ortogonale, ρ la densità del fluido, p la pressione esercitata sul fluido e t il tempo. Il primo membro rappresenta la variazione rispetto al tempo della quantità di moto per unità di volume. Tale variazione è dovuta a tre effetti: il flusso dovuto al campo di moto del fluido (i primi due termini al secondo membro), detto termine di convezione, le forze di pressione (terzo termine) e la dissipazione (quarto e quinto termine). La non linearità (dovuta al termine di convezione), causa principale della difficoltà di risoluzione per casi non idealizzati [168], ha spinto alcuni ricercatori a tentare un approccio alternativo allo studio della fluidodinamica basato sugli AC che ha preso il nome di gas reticolari.

Gas Reticolari

L'idea base dei gas reticolari è modellare un fluido attraverso un sistema di particelle ognuna delle quali è vincolata a muoversi, con velocità costante, solo lungo le direzioni di un reticolo discreto. Le leggi locali sono definite in modo da garantire

l'invarianza del numero di particelle (conservazione della massa) e la conservazione della quantità di moto.

I primi a proporre un approccio di questo tipo sono stati, nel 1976, Hardy, Pomeau e de Pazzis [80]. Il modello HPP (dalle iniziali dei tre ricercatori) è basato su una griglia quadrata in cui ciascun nodo è collegato a quattro vicini (vicinato di von Neuman, figura 3.3a). Ogni nodo può contenere da un minimo di zero a un massimo di quattro particelle di massa unitaria, ognuna delle quali può spostarsi soltanto lungo una delle direzioni del reticolo con velocità costante, pari a 1 (cioè, ogni particella può spostarsi in un passo dell'AC dalla cella in cui sitrova in una cella adiacente). Poichè si è supposta sia la massa che la velocità unitarie, la quantità di moto di una particella può essere definita semplicemente dal verso (i versi consentiti sono Sud, Ovest, Nord ed Est). Pertanto, le possibili configurazioni per ogni nodo dell'HPP sono $2^4 = 16$. Per esempio la configurazione 0000 indica l'assenza di particelle, la configurazione 0001 indica la presenza di una particella che si sta spostando verso Est, la configurazione 0010 indica la presenza di una particella che si sta spostando verso Nord, e coì via.

Quando due particelle con velocità opposte si trovano nella stessa cella (configurazioni 0101 e 1010), si verifica un urto e le particelle vengono deviate di un angolo retto. In tutti gli altri casi, compresi i casi in cui i cammini si incrociano (per esempio nella conigurazione 1111), le particelle proseguono il loro cammino senza deviazioni.

Quando il numero di particelle è sufficientemente grande, su scala macroscopica l'impressione è quella di un fluido continuo. L'AC HPP si è dimostrato capace di modellare correttamente fenomeni fluidodinamici come la propagazione di un'onda, ma il suo comportamento risulta poco realistico nella simulazione di fluidi turbolenti. Il problema principale è dovuto al vincolo reticolare. Quest'ultimo è, infatti, responsabile dei così detti invarianti spuri [168], cioè invarianti che non hanno corrispondenti nel continuo. In pratica l'HPP, oltre a conservare la massa e la quantità di moto, conserva anche la quantità di moto lungo le righe e le colonne del reticolo, come si può dedurre facilmente dall'analisi delle regole di transizione. A causa di questo eccesso d'invarianti, il moto delle particelle risulta fortemente anisotropo e il fluido non è sufficientemente libero di muoversi e diffondere la sua quantità di moto in tutte le direzioni.

Nel 1986, però, Frish, Hasslacher e Pomeau [69], intuirono che l'impiego di un reticolo a maglia esagonale regolare (figura 3.1c), in sostituzione di quello a maglia quadrata, potesse bastare a ristabilire l'isotropia perduta. Nel modello FHP (ancora dalle iniziali dei tre ricercatori), ogni cella è collegata alle sei celle

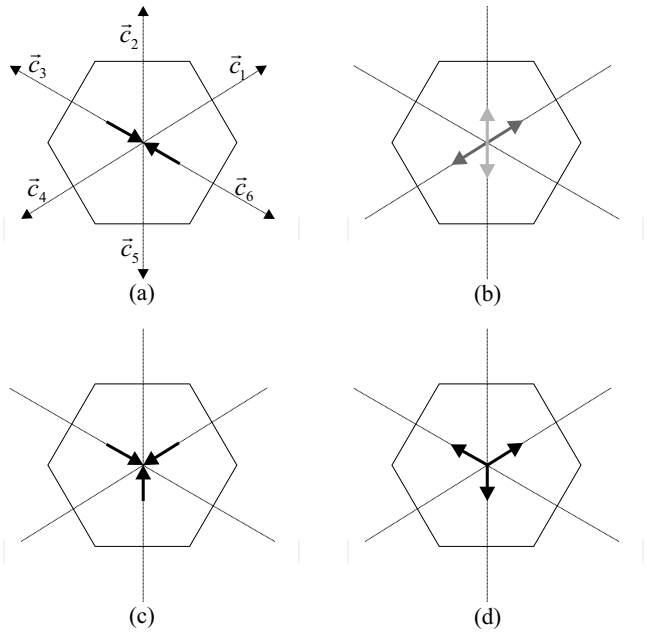


Figura 3.12: Regole di collisione del modello FHP. (a) e (b) descrivono, rispettivamente, la configurazione della cella prima e dopo l'urto nel caso di due particelle; si noti che in questo caso la regola di transizione è probabilistica: le particelle possono, cioè, deviare verso destra (configurazione in grigio chiaro) o sinista (configurazione in grigio scuro) con la stessa probalibità). (c) e (d) descrivono, rispettivamente, la configurazione della cella prime e dopo l'urto nel caso di tre particelle. Le sei possibili direzioni delle particelle, $c_1, c_2, \ldots, c_6$, sono evidenziate in (a).

vicine secondo angoli di 60 gradi (figura 3.3c). Potendo le particelle muoversi lungo tre direzioni invece che due, il numero di possibili stati della cella passa da 2^4 a 2^6, pur rimanendo le leggi di transizione sostanzialmente equivalenti a quelle del modello HPP, con la differenza però che nel modello FHP sono possibili anche urti che coinvolgono contemporaneamente tre particelle.

Sia $n_i(\overrightarrow{r}, t)$ il numero di particelle (0 o 1) entranti nella cella, individuata dal vettore $\overrightarrow{r}$ al passo t lungo la direzione $\overrightarrow{c_i}$ ($i = 1, 2, \ldots, 6$) (figura 3.12). Sia, inoltre, τ il tempo corrispondente a un passo di calcolo del modello FHP e λ la distanza tra due celle vicine del reticolo esagonale. Le velocità $\overrightarrow{v_i}$ delle particelle lungo le direzioni $\overrightarrow{c_i}$ sono definite nel seguente modo:

$$\overrightarrow{v_i} = \frac{\lambda}{\tau}\overrightarrow{c_i}$$

La funzione di transizione della cella può essere dedotta dalle regole di collisione,

illustrate in figura 3.12, considerando il caso in cui non si verificano urti e i due casi in cui, invece, hanno luogo collisioni tra due o tre particelle:

Nessuna collisione. Se non si verifica alcuna collisione le particelle proseguono il loro cammino lungo l'originaria direzione reticolare, senza alcuna deviazione. Pertanto vale la seguente regola d'aggiornamento:

$$n_i(\overrightarrow{r} + \lambda\overrightarrow{c_i}, t+\tau) = n_i(\overrightarrow{r}, t)$$

cioè, ogni particella che al tempo t si trovi nella cella individuata dal vettore $\overrightarrow{r}$, al passo successivo, $t+\tau$, sarà nella cella adiacente, lungo la stessa direzione, individuata dal vettore $\overrightarrow{r} + \lambda\overrightarrow{c_i}$.

Collisione tra due particelle. Se solo n_i ed n_{i+3} valgono 1 nella cella $\overrightarrow{r}$, come illustrato in figura 3.12a-b, si verifica una collisione. La condizione

$$D_i = n_i n_{i+3}(1-n_{i+1})(1-n_{i+2})(1-n_{i+4})(1-n_{i+5}) = 1$$

identifica tale urto. Pertanto il numero di particelle che permangono lungo la direzione $\overrightarrow{c_i}$ dopo l'urto è dato da:

$$n_i - D_i$$

Tuttavia, una nuova particella può comparire nella direzione $\overrightarrow{c_i}$ per effetto delle collisioni tra n_{i-1} ed n_{i+2} oppure tra n_{i+1} ed n_{i+4}. Il numero di particelle create nella direzione $\overrightarrow{c_i}$ può essere, pertanto, dedotto dalla seguente formula:

$$qD_{i-1} + (1-q)D_{i+1}$$

essendo $q = q(\overrightarrow{r}, t)$ una variabile booleana random che vale 0 se dopo l'urto le particelle deviano verso sinistra, 1 se deviano verso destra. Pertanto, il numero complessivo di particelle lungo $\overrightarrow{c_i}$ è dato da:

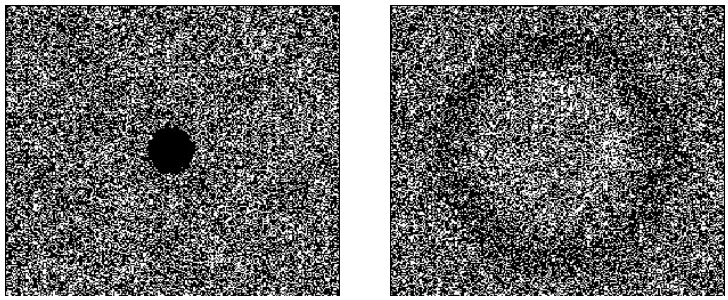
$$n_i - D_i + qD_{i-1} + (1-q)D_{i+1}$$

Collisione tra tre particelle. La condizione

$$T_i = n_i n_{i+2} n_{i+4}(1-n_{i+1})(1-n_{i+3})(1-n_{i+5}) = 1$$

identifica un urto tra tre particelle (figura 3.12c-d). L'effetto è quello di deviare le particelle di un angolo di 180 gradi. Pertanto il numero di particelle che rimangono lungo la direzione $\overrightarrow{c_i}$ dopo l'urto è dato da:

$$n_i - T_i$$



(a) (b)

Figura 3.13: Dinamica di un'onda (b) nel modello FHP dovuta alla diffusione di una concentrazione di particelle nella zona centrale del reticolo (a).

Tuttavia, un urto tra le particelle n_{i+1}, n_{i+3} ed n_{i+5} determina la comparsa di una nuova particella lungo la direzione $\overrightarrow{c_i}$, condizione che si esprime nel seguente modo:

$$T_{i+3}$$

Il numero complessivo di particelle nella direzione $\overrightarrow{c_i}$ per effetto di collisioni tra tre particelle è dato, dunque, da:

$$n_i - T_i + T_{i+3}$$

Pertanto la dinamica del modello FHP lungo ogni direzione del reticolo è definita dalla seguente legge:

$$n_i(\overrightarrow{r} + \lambda\overrightarrow{c_i}, t + \tau) = n_i(\overrightarrow{r}, t) + \Omega_i(\overrightarrow{r}, t)$$

in cui il termine $\Omega_i(\overrightarrow{r}, t) = -D_i + qD_{i-1} + (1-q)D_{i+1} - T_i + T_{i+3}$ rappresenta gli effetti delle collisioni tra le particelle; Ω_i è detto, infatti, termine di collisione.

Con queste poche e, tutto sommato, semplici modifiche (l'introduzione del reticolo esagonale e le nuove regole di collisione), l'FHP ha risolto il problema degli invarianti spuri e si è dimostrato capace di riprodurre correttamente fenomeni fluidodinamici complessi. Inoltre, Chopard e Droz [30] hanno dimostrato che l'introduzione di particelle con velocità nulla nelle versioni successive del modello FHP [62] consente di derivare dalle regole di collisione le equazioni di Navier-Stokes per fluidi non comprimibili. La figura 3.13 illustra un'applicazione del modello FHP.

L'isotropia, cioè l' equivalenza di tulle le direzioni del reticolo, gioca dunque un ruolo estremamente importante nella modellizzazione del comportamento delle

particelle del fluido. Questo aspetto costituì un problema apparentemente insormontabile nel passaggio alla terza dimensione, che maggiormente interessa sul piano delle applicazioni pratiche. Infatti non esiste alcun solido elementare in grado di riempire in maniera congruente (cioè senza buchi) lo spazio, garantendo al tempo stesso l'isotropia [168]. Nel 1987 Frish, d'Humières e Lallemand, notarono però che già nello spazio a quattro dimensioni un solido con le proprietà richieste esiste. Si tratta di un ipercubo (la generalizzazione di un cubo) a facce centrate, su cui si è basata la costruzione del reticolo FCHC (Face Centered Hyper Cube) [68]. Il passaggio alla terza dimensione, benchè ineccepibile dal punto di vista teorico, pose tuttavia pesanti problemi in termini computazionali. Nell'FCHC si hanno 12 possibili direzioni e, di conseguenza, il numero di possibili stati diventa 2^{24} (oltre 16 milioni). Così, la tabella di transizione che definisce l'esito di tutte le possibili configurazioni della cella veniva a essere di ben 48 Mbyte. Pur non considerando il fatto che nel 1987 una tale quantità era considerata decisamente alta, il problema più pesante era che una tabella di transizione così grande richiedeva tempi d'accesso inaccettabili e, per questo, molti studi si occuparono di ridurne le dimensioni.

Modelli di Boltzmann su reticolo

Contemporaneamente alla ricerca di tavole di collisione ridotte, si è sviluppata una soluzione alternativa, oggi nota come metodo di Boltzmann su reticolo, proposta per la prima volta da McNamara e Zanetti [125] dell'Università di Chicago e, contemporaneamente, da Higuera e Jimenez del Politecnico di Madrid [83].

Un vantaggio non trascurabile dei modelli di Boltzmann su reticolo, rispetto ai Gas Reticolari, è la più elevata efficienza computazionale, visto che in questi ultimi le quantità d'interesse non sono più le singole particelle ma la loro densità. Così, il numero di componenti del sistema si riduce sensibilmente e non è più necessaria la fase del calcolo di medie spazio-temporali per la determinazione delle quantità fisiche d'interesse del sistema: la densità del fluido, ρ, la quantità di moto, $\rho\,\overrightarrow{u}$, sono note in ogni cella del reticolo.

Nella sua forma più semplice [152, 29], la dinamica di un modello di Boltzmann su reticolo può essere descritta nel seguente modo:

$$f_i(\overrightarrow{r} + \tau\overrightarrow{v_i}, t+\tau) - f_i(\overrightarrow{r}, t) = \Omega_i(f_i(\overrightarrow{r}, t)) = \frac{1}{\xi}\left(f_i^{(eq)}(\overrightarrow{r}, t) - f_i(\overrightarrow{r}, t)\right)$$

dove $f_i(\overrightarrow{r}, t)$ rappresenta la densità di particelle che al tempo t si trovano nella cella $\overrightarrow{r}$ con velocità $\overrightarrow{v_i}$; $f_i^{(eq)}(\overrightarrow{r}, t)$ rappresenta la, così detta, distribuzione d'equilibrio

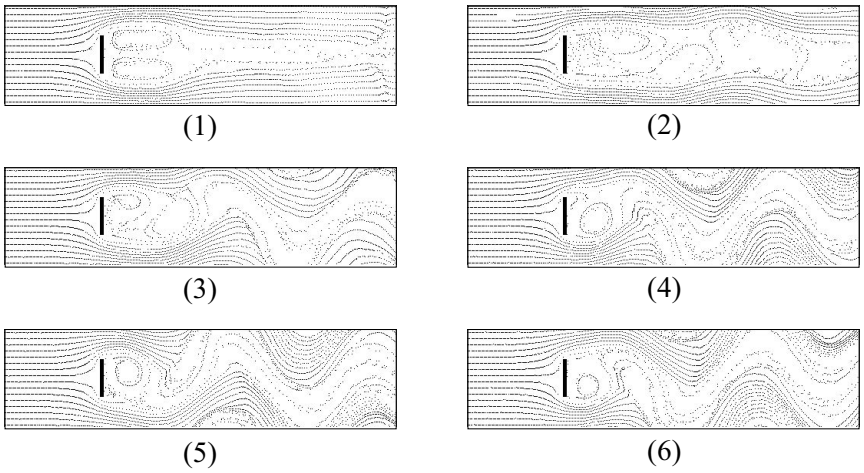


Figura 3.14: Simulazione di un flusso intorno a una lamina in un modello di Boltzmann su reticolo di 512x128 celle e $\xi = 1$. Le figure da 1 a 6 illustrano l'evoluzione del sistema.

locale; ξ, detto termine di rilassamento, rappresenta il numero di passi di calcolo necessari affichè si raggiunga l'equilibrio nel contesto locale del vicinato.

Le leggi che determinano la dinamica nei modelli di Boltzmann su reticolo riducono, dunque, le condizioni di non equilibrio nel contesto locale del vicinato. La funzione $f_i^{(eq)}(\overrightarrow{r}, t)$ specifica le condizioni d'equilibrio locale del sistema in funzione della densità, $\rho = \sum f_i$, e della quantità di moto, $\rho\overrightarrow{u} = \sum f_i \overrightarrow{v_i}$, del fluido nella cella. Il parametro ξ esprime, invece, la dipendenza del sistema dalla viscosità, ν, del fluido. Vale, infatti la seguente relazione: $\nu = K(\xi - 1/2)$, dove K è una costante che dipende dalla geometria del reticolo [31]. In tal modo la viscosità diviene un parametro esplicito del modello, a differenza di quanto accade nei Gas Reticolari in cui essa dipende implicitamente dalla specificazione delle leggi che regolano gli urti tra le particelle.

I modelli di Boltzmann su reticolo coinvolgono, dunque, quantità esprimibili non più come numeri interi (come nel caso delle singole particelle nei gas reticolari) ma come numeri reali. Se da una parte questo può rappresentare un problema dal punto di vista della stabilità numerica (intesa come propagazione dell'errore, tipica dei modelli basati sull'elaborazione di valori in virgola mobile), dall'altra offre una maggiore flessibilità nella calibratura del modello dato che è possibile esplicitare, come nel caso della viscosità, alcuni importanti parametri del sistema. Inoltre, come per il modello FHP, anche per i modelli di Boltzmann su reticolo è stata dimostrata l'equivalenza con le equazioni di Navier-Stokes per fluidi incomprimibili

[30]. La figura 3.14 illustra la dinamica del modello BKG [152] nella simulazione di un flusso intorno a una lamina.

Studi teorici e applicazioni dei gas reticolari e del metodo di Boltzmann su reticolo sono descritti in Rothman e Zaleski [157], in Luo [117], in Chopard e Masselot [32], in Bernaschi et al. [21], in Succi [169] e in Wolfram [188].

Capitolo 4

Fenomeni macroscopici e Automi Cellulari

4.1 Introduzione

Le leggi della Fisica sono, in ultima istanza, fondate su principi di conservazione, quali la conservazione della massa, della quantità di moto e dell'energia [30]. Nel passato, sino all'avvento degli elaboratori elettronici, la possibilità di descrivere i fenomeni in termini di equazioni differenziali, le cui soluzioni analitiche forniscono lo stato del sistema in ogni punto del continuum spazio-temporale, ha segnato la differenza tra scienza forte (altamente predittiva) e scienza debole (puramente descrittiva). Purtroppo, però, come nel caso delle equazioni di Navier-Stokes in fluidodinamica, per molti sistemi non è ancora nota la soluzione analitica, il chè costituisce una grossa limitazione nello studio quantitativo dei sistemi d'interesse [168].

Tentativi di modellizzazione quantitativa di fenomeni naturali complessi attraverso metodi numerici approssimati (comunemente basati sulla discretizzazione dello spazio e del tempo) si sono principalmente sviluppati grazie alla crescita della potenza dei calcolatori elettronici . Questi metodi (per esempio per l'analisi della stabilità dei pendii [115, 129, 185, 24]) hanno permesso di estendere la classe di problemi che possono essere risolti in termini di sistemi di equazioni differenziali. Tuttavia, alcuni di essi risultano ancora difficilmente trattabili, altri addirittura non lo sono affatto.

Per studiare e risolvere tali problemi, la comunità scientifica internazionale ha dovuto superare le difficoltà inerenti la risoluzione dei sistemi di equazioni differen-

ziali tramite varie tecniche alternative [172]. Contemporaneamente allo sviluppo di metodi approssimati per la risoluzione numerica di sistemi di equazioni differenziali [153], si sono infatti sviluppati e consolidati nuovi metodi numerici che si fondano sui principi del calcolo parallelo. Entrambi gli approcci, quello basato sulle equazioni differenziali e quello basato sui nuovi modelli di calcolo parallelo, si basano, tuttavia, sui concetti comuni di modellizzazione e simulazione.

4.2 Modellizzazione con automi cellulari

Gli automi cellulari hanno introdotto un approccio radicalmente nuovo nella modellizzazione di fenomeni complessi che evolvono sulla base di leggi locali. Nel capitolo precedente sono stati presentati i Gas Reticolari e i modelli di Boltzmann su reticolo, che per primi hanno evidenziato le potenzialità degli AC nella modellizzazione e simulazione di sistemi fisici complessi.

Mentre i Gas Reticolari sono stati sempre considerati AC a tutti gli effetti (reticolo discreto, numero finito di stati per la cella, funzione di transizione esprimibile in termini di look-up table), i modelli di Boltzmann su reticolo sono stati visti, per un certo periodo, come "quasi automi" [168]. Infatti, nei modelli di Boltzmann su reticolo lo stato della cella rappresenta la densità delle particelle nella corrispondente regione di spazio e quindi può variare in un intervallo continuo di valori. Tuttavia la continuità è un problema solo dal punto di vista formale; nei casi pratici le variabili reali hanno un numero finito di cifre significative e, di conseguenza, un insieme finito di valori permessi. In altri termini l'insieme dei valori permessi può essere estremamente grande ma in ogni caso è finito. Così i modelli di Boltzamnn su reticolo possono essere considerati AC a tutti gli effetti.

Sia i Gas Reticolari che i modelli di Boltzmann su reticolo sono stati applicati alla simulazione della turbolenza dei fluidi. Tuttavia il loro ambito d'applicabilità non include, in generale, fenomeni macroscopici, come l'evoluzione di colate laviche o detritiche, che evolvono in contesti puramente tridimensionali.

Tuttavia, negli ultimi decenni sono stati proposti numerosi modelli empirici basati sugli AC per la simulazione di fenomeni macroscopici complessi. Tra i più recenti Smith [167], Murray e Paola [131, 132] e D'Ambrosio et al. [47] hanno proposto modelli per la simulazione dell'erosione del suolo; Barca et al. [17], Crisci et al. [56, 41, 38, 40] e Miyamoto e Sasaki [127] hanno proposto modelli per la simulazione di flussi lavici; Crisci et al. [40] hanno proposto un modello per la simulazione di flussi piroclastici; Sassa [162], Segre e Deangeli [163], Malamud e

Turcotte [118, 119], Clerici e Perego [34], Avolio et al. [10, 9], Di Gregorio et al. [58], D'Ambrosio et al. [50, 51, 49, 52] e Iovine et al. [88] hanno proposto modelli per la simulazione di flussi detritici.

4.2.1 Un metodo empirico per la modellizzazione di fenomeni macroscopici con Automi Cellulari

In generale, i fenomeni macroscopici si esplicano su scale spaziali decisamente più grandi di quelle che caratterizzano le applicazioni dei Gas Reticolari e dei modelli di Boltzmann su reticolo. In molti casi l'approccio microscopico fallisce banalmente per l'insufficiente qualità dei dati. Nella modellizzazione e simulazione di fenomeni di flusso su una superficie (per esempio sul versante di una montagna) non è possibile prescindere dalla topografia della regione su cui il fenomeno evolve. Generalmente le mappe topografiche sono interpretazioni di foto aeree o di immagini satellitari e il loro dettaglio difficilmente scende al disotto dell'ordine dei metri. In altri termini quella che oggi è ritenuta una mappa topografica digitale dettagliata fornisce le quote dell'area d'interesse tramite un reticolo quadrato di quote in cui la distanza tra due punti vicini e dell'ordine di uno-due metri. Una tale mappa è detta DEM (Digital Elevation Model).

Anche se in linea di principio l'AC, così come definito nel capitolo precedente, è senz'altro sufficiente alla modellizzazione e simulazione di fenomeni naturali che evolvono su scala macroscopica, la loro complessità suggerisce un'estensione dell'originario paradigma. Le considerazioni che seguono introducono un'estensione della definizione di AC che meglio si adatta alla modellizzazione di fenomeni macroscopici complessi.

Parametri

Innanzitutto, deve essere univocamente determinata una corrispondenza spaziale tra il reticolo discreto dell'AC e la regione in cui evolve il fenomeno; allo stesso modo, deve essere fissato il tempo equivalente a un passo di calcolo dell'AC. Una volta determinati, la dimensione della cella e il "clock" dell'AC sono definiti come "parametri", dato che i loro valori sono assunti costanti per tutto l'arco della simulazione. Nel seguito si indicherà con p_c la misura del lato della cella (in caso AC con reticolo a maglia quadrata) o con p_a la misura dell'apotema (in caso di reticoli di a maglia esagonale regolare) e con p_s il tempo corrispondente a un

passo dell'AC. Essi costituiscono, insieme ad altri eventuali parametri solitamente necessari ai fini della simulazione, l'insieme P dei parametri dell'AC.

Si noti tuttavia che, nel contesto del metodo empirico presentato in questo capitolo, a differenza della dimensione della cella che è generalmente imposta *a priori* dal dettaglio della mappa digitale dell'area in cui evolve il fenomeno, una stima esatta del tempo corrispondente a un passo di calcolo dell'AC può essere data, in generale, solo *a posteriori* dall'analisi del comportamento globale del sistema. Il valore del parametro p_c può dipendere, infatti, significativamente dal valore di alcuni altri parametri del modello, per esempio dal parametro p_r, discusso nel paragrafo 4.2.4.

Sottostati

Lo stato della cella deve tenere in considerazione tutte le caratteristiche che sono ritenute rilevanti per l'evoluzione del sistema. Per comodità e maggior chiarezza, nell'approccio considerato ogni caratteristica corrisponde a un "sottostato", i cui valori devono formare un insieme finito; $S = \{Q_1, Q_2, \ldots, Q_n\}$ è l'insieme dei sottostati dell'AC. Come nel caso dei modelli di Boltzmann su reticolo, la necessità di modellare proprietà caratterizzate da valori continui non pone sostanziali problemi (si veda la sezione 4.2). L'insieme Q di tutti i possibili valori dello stato della cella è, dunque, espresso come prodotto cartesiano degli elementi di S:

$$Q = Q_1 \times Q_2 \times \ldots \times Q_n$$

Il valore di un sottostato è sempre considerato costante all'interno della cella; tale valore è cioè rappresentativo di tutta la corrispondente porzione di spazio.

Processi elementari

La funzione di transizione σ dell'AC deve tenere in considerazione tutti i processi (fisici, chimici, ecc.), responsabili del cambiamento dei valori dello stato della cella, che sono ritenuti rilevanti per l'evoluzione del sistema. Così come l'insieme degli stati Q è stato decomposto nei sottostati $Q_1, Q_2, \ldots, Q_n$, anche la funzione di transizione σ è decomposta in "processi elementari". A loro volta, i processi elementari si suddividono in "trasformazioni interne", $T_1, T_2, \ldots, T_p$, e "interazioni locali", $I_1, I_2, \ldots, I_q$.

Le trasformazioni interne determinano il cambiamento dei valori dei sottostati della cella dovuti esclusivamente all'interazioni tra i sottostati della cella stessa

oppure semplicemente allo scorrere del tempo. In altri termini le trasformazioni interne non dipendono dallo stato delle celle del vicinato ma solo dallo stato della cella centrale. Le interazioni locali determinano, invece, il cambiamento dei valori dei sottostati della cella dovuti all'interazione con le celle del vicinato.

Per ogni trasformazione interna T_i $(i = 1, 2, \ldots, p)$ è definita una funzione

$$\sigma_{T_i} : Q_{T_i} \rightarrow Q'_{T_i}$$

dove Q_{T_i} e Q'_{T_i} sono prodotti cartesiani di elementi di S.

Allo stesso modo, per ogni interazione locale I_j $(j = 1, 2, \ldots, q)$ è definita una funzione

$$\sigma_{I_j} : Q_{I_j}^m \rightarrow Q'_{I_j}$$

dove Q_{I_j} e Q'_{I_j} sono prodotti cartesiani di elementi di S ed m è il numero di celle del vicinato.

L'evoluzione del sistema è ottenuta applicando, secondo un ordine che dipende dalle caratteristiche del fenomeno modellato, le trasformazioni interne e le interazioni locali a ogni cella dell'AC. Questa assunzione, tuttavia, non è giustificabile *a priori* e deve essere verificata empiricamente nella fase di simulazione.

Influenze esterne

In alcuni casi è necessario considerare qualche tipo di input dal "mondo esterno" che tenga conto di influenze non descrivibili in termini di leggi locali dell'AC. Di conseguenza la definizione del modello può prevedere la definizione di una o più funzioni speciali e/o addizionali. Tipici esempi d'influenze esterne sono l'emissione di lava dai crateri in modelli di simulazione di flussi lavici o l'innesco di fenomeni franosi in modelli di simulazione di flussi detritici.

Dimensione della cella, passo temporale e relazione di vicinanza

La scelta dei valori dei parametri p_c e p_s deve essere fatta in relazione ai processi elementari considerati. Nell'operare tale scelta, a causa dell'elevata complessità dei fenomeni considerati, è possibile incorrere in alcuni problemi: la dimensione più opportuna per la cella potrebbe essere diversa per diversi processi elementari; inoltre alcune interazioni locali potrebbero richiedono clock più piccoli rispetto ad altri processi elementari; infine la relazione di vicinanza potrebbe variare da un'interazione locale all'altra. In questi casi si procede nel seguente modo: la dimensione della cella e il clock dell'AC sono scelti tra i valori più piccoli tra

quelli considerati e, nel caso siano definiti più vicinati, la loro unione è scelta come vicinato dell'AC.

4.2.2 Estensione della definizione di Automa Cellulare per la modellizzazione di fenomeni macroscopici

Definizione 4.2.1. Un Automa Cellulare per la modellizzazione di fenomeni macroscopici complessi è formalmente definito nel seguente modo:

$$A =< \mathbb{Z}^d, Q, P, X, \sigma, E, \gamma >$$

dove:

$\mathbb{Z}^d = \{i \equiv (i_1, i_2, \ldots, i_d) \mid i_k \in \mathbb{Z}\ \forall k = 1, 2, ..., d\}$ è l'insieme dei punti del reticolo d-dimensionale che definisce lo spazio cellulare dell'AC; $\mathbb{Z}$ è l'insieme dei numeri interi;

$Q = Q_1 \times Q_2 \times \ldots \times Q_n$ è l'insieme finito degli stati dell'automa elementare, espresso come prodotto cartesiano dei sottostati $Q_1, Q_2, \ldots, Q_n$;

$P = \{p_1, p_2, \ldots, p_l\}$ è l'insieme finito dei parametri dell'AC;

$X = \{\xi_0, \xi_1, \ldots, \xi_{m-1}\}$ è l'insieme finito degli m vettori d-dimensionali

$$\xi_j = \{\xi_{j_1}, \xi_{j_2}, \ldots, \xi_{j_d}\}$$

che definiscono l'insieme

$$V(X, i) = \{i + \xi_0, i + \xi_1, \ldots, i + \xi_{m-1}\}$$

delle coordinate delle celle vicine alla generica cella i di coordinate $(i_1, i_2, \ldots, i_d)$. X è detto indice o relazione di vicinanza;

$\sigma : Q^m \to Q$ è la funzione di transizione locale dell'automa elementare. Essa è decomposta nelle trasformazioni interne, $\sigma_{T_1}, \sigma_{T_2}, \ldots, \sigma_{T_p}$ e nelle interazioni locali, $\sigma_{I_1}, \sigma_{I_2}, \ldots, \sigma_{I_q}$. Per ogni interazione locale σ_{I_k} può essere definita una particolare relazione di vicinanza X_{I_k} $(k = 1, 2, \ldots, q)$. In tal caso vale $X = X_{I_1} \cup X_{I_2} \cup \ldots \cup X_{I_q}$;

$E = E_1 \cup E_2 \cup \ldots \cup E_s \subseteq \mathbb{Z}^d$ è l'insieme delle celle di $\mathbb{Z}^d$ soggette a influenze esterne;

$\gamma = \{\gamma_1, \gamma_2, \ldots, \gamma_s\}$ è l'insieme finito delle funzioni che definiscono gli input esterni per l'AC; $\gamma_i : \mathbb{N} \times E_i \times Q \to Q$ $(i = 1, 2, \ldots, s)$, essendo $\mathbb{N}$ l'insieme dei numeri naturali, rappresentanti i passi dell'AC.

4.2.3 Modellizzazione di flussi di superficie

I flussi di superficie (colate laviche, flussi detritici, e altri) appartengono alla classe di problemi naturalmente descrivibili in termini d'interazioni locali. Immaginando di discretizzare la superficie (per esempio il versante di una montagna) su cui evolve il fenomeno, la dinamica del sistema può essere, infatti, descritta in termini di flussi di certe quantità (lava, detrito, ecc.) tra le celle dell'AC.

L'approccio metodologico sopra illustrato permette, inoltre, di considerare la terza dimensione (quota) come una proprietà (sottostato) della cella, consentendo l'applicazione di AC bidimensionali a fenomeni tipicamente tridimensionali; questo è il caso di alcuni fenomeni di superficie, tra cui i flussi detritici. In considerazione del fatto che la misura della cella è costante in tutto lo spazio cellulare, è infine possibile riferirsi alle caratteristiche (sottostati) della cella, tipicamente espresse in termini di volume (per esempio al volume di lava), in termini di spessore.

Queste assunzioni permettono di adottare una semplice ma efficace strategia che consiste nel calcolo dei flussi uscenti (di detrito, di lava, ecc., anch'essi in termini di spessore) dalla cella centrale verso le celle vicine al fine di minimizzare le condizioni di non equilibrio del sistema.

Tale assunzione è esplicitata, nel prossimo paragrafo, tramite la definizione dell'"algoritmo di minimizzazione delle differenze", insieme alle sue proprietà e a un esempio d'applicazione.

4.2.4 L'algoritmo di minimizzazione

L'algoritmo di minimizzazione delle differenze, proposto da Di Gregorio e Serra [59], riduce le condizioni di non equilibrio minimizzando le differenze di certe quantità (per esempio gli spessori lava, di detrito o l'energia) tra la cella centrale e le sue vicine.

Sia $n = \#X$ il numero di celle del vicinato; l'algoritmo di minimizzazione è basato sulle seguenti assunzioni:

- il contenuto della cella centrale è suddiviso in due quantità: la parte inamovibile, $u(0)$, e la parte mobile, m;

- soltanto m può essere distribuita alle celle adiacenti; essa è definita come

$$m = \sum_{i=0}^{n-1} q_o(0, i) \tag{4.1}$$

  dove la notazione $q_o(x, y)$ rappresenta il flusso che si sposta dalla cella x alla cella y; quindi, $q_o(0, i)$ rappresenta la porzione di m che si sposta dalla cella centrale verso l'i-esima vicina ($i = 1, 2, \ldots, n-1$); $q_o(0, 0)$ rappresenta la porzione di m che non è distribuita. Si noti che tale definizione garantisce il principio di conservazione della massa per la quantità distribuibile m;

- le quantità $u(i)$ ($i = 1, 2, \ldots, n-1$) nelle celle adiacenti sono considerate inamovibili;

- $c(i) = u(i) + q_o(0, i)$ ($i = 0, 1, \ldots, n-1$) rappresenta la quantità contenuta nelle celle del vicinato dopo la distribuzione del flusso; c_{min} rappresenta il minimo valore dei $c(i)$;

L'algoritmo di minimizzazione procede nel seguente modo:

1. si calcola la media
$$a = \frac{m + \sum_{i \in A} u(i)}{\#A}$$
   dove A è l'insieme delle celle non eliminate (che possono cioè ricevere un flusso); al primo passo $\#A = \#X = n$;

2. le celle per le quali $u(i) \geq a$ ($i = 0, 1, \ldots, n-1$) sono eliminate dalla distribuzione e, quindi, dal calcolo della nuova media;

3. i punti 1 e 2 sono iterati fino a quando nessuna cella è eliminata; il flusso dalla cella centrale verso l'i-esima vicina non eliminata è calcolata come la differenza tra $u(i)$ e l'ultima media:
$$q_o(0, i) = a - u(i)$$
   per le celle eliminate, invece, $q_o(0, i) = 0$.

La figura 4.1 illustra un esempio d'applicazione dell'algoritmo di minimizzazione delle differenze nel caso di un AC bidimensionale con celle quadrate e vicinato di von Neumann.

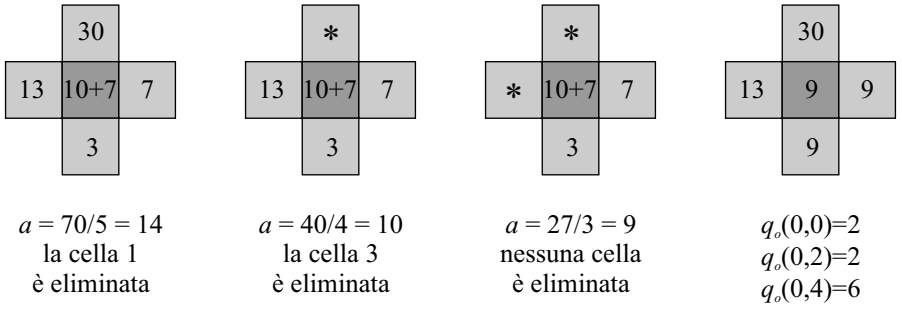


Figura 4.1: Esempio d'applicazione dell'algoritmo di minimizzazione delle differenze nel caso di un AC bidimensionale con celle quadrate e vicinato di von Neumann. Per convenzione si identifica la cella centrale come cella 0, la Nord come cella 1, la Est come cella 2, la Ovest come cella 3 e la Sud come cella 4. Il simbolo * indica l'eliminazione della cella dal calcolo della media e dalla distribuzione del flusso. Nella cella centrale sono considerate due quantità, una inamovibile, $u(0) = 7$ e una movibile $m = 10$. L'algoritmo di minimizzazione determina i flussi $q_o(0, i)$ $(i = 0, 1, \ldots, 4)$ dalla cella centrale verso le vicine.

Teorema 4.2.1. *(Di Gregorio e Serra, 1999) L'algoritmo sopra descritto minimizza la seguente espressione:*

$$\sum_{i=0}^{n-1}(c(i) - c_{min})$$

Dimostrazione. Tenendo presente che $c(i) = u(i) + q_o(0, i)$ e che $q_o(0, i) = a - u(i)$ $(i = 0, 1, \ldots, n-1)$, se $u(i) \geq a$, allora

$$q_o(0, i) = 0 \Rightarrow c(i) = u(i) \geq a$$

mentre se $u(i) < a$, allora

$$q_o(0, i) = a - u(i) \Rightarrow c(i) = a$$

Di conseguenza si ha:

$$c_{min} = a$$

Inoltre, ogni variazione nei valori dei flussi determina un valore per c_{min} minore della media a. Infatti, se si considera un incremento nel valore del flusso verso una generica cella i, $q'_o(0, i) > q_o(0, i)$, dopo la distribuzione del flusso risulta:

$$c'(i) = u(i) + q'_o(0, i) > u(i) + q_o(0, i) = c(i)$$

Tuttavia, in base all'equazione 4.1, l'aumento del flusso verso la cella i determina la diminuzione di almeno un altro flusso, diciamo verso al cella k ($k \neq i$), $q'_o(0,k) < q_o(0,k)$. Pertanto, dopo la distribuzione del flusso, nella vicina k risulta:

$$c'(k) = u(k) + q'_o(0,k) < u(k) + q_o(0,k) = c(k)$$

In tal modo, una qualsiasi variazione nel valore dei flussi calcolati tramite l'algoritmo di minimizzazione è tale che

$$c'_{min} < a$$

Di conseguenza:

$$\sum_{i=0}^{n-1}(c(i) - c_{min}) = \sum_{i=0}^{n-1}(u(i) + q_o(0,i) - c_{min}) = \sum_{i=0}^{n-1} u(i) + m - n \cdot c_{min} \qquad (4.2)$$

per il caso in cui i flussi non siano stati variati, e

$$\sum_{i=0}^{n-1}(c'(i) - c'_{min}) = \sum_{i=0}^{n-1}(u(i) + q'_o(0,i) - c'_{min}) = \sum_{i=0}^{n-1} u(i) + m - n \cdot c'_{min} \qquad (4.3)$$

per il caso in cui i flussi siano stati variati. Dalle equazioni 4.2 e 4.3 si ricava

$$\sum_{i=0}^{n-1}(c(i) - c_{min}) < \sum_{i=0}^{n-1}(c'(i) - c'_{min})$$

□

Poichè, in generale, in relazione alla dimensione della cella, la condizione d'equilibrio non deve essere ottenuta in un unico passo di calcolo, è necessario considerare un fattore moltiplicativo di "rallentamento", $p_r \in (0,1]$, che "smorzi" l'entità dei flussi. Quando p_r è definito costante nello spazio e nel tempo, esso è considerato dal punto di vista formale come un parametro dell'AC. Si noti che il valore del parametro p_r influenza in maniera diretta il clock, p_s, dell'AC poichè determina il tempo di rilassamento del sistema verso l'equilibrio: tanto maggiore è il valore del parametro di rallentamento, tanto più piccolo deve essere il tempo corrispondente a un passo di calcolo.

L'applicazione simultanea del principio di minimizzazione alle situazioni di non equilibrio a tutte le celle dall'AC dà luogo all'emergenza dell'equilibrio globale di tutto il sistema.

4.2.5 Un esempio d'applicazione del nuovo approccio metodologico alla simulazione di flussi lavici

Il comportamento dei flussi di lava appartiene alla classe dei fenomeni complessi che sono usualmente difficili da simulare usando metodi tradizionali basati su sistemi di equazioni differenziali. SCIARA (*Simulation by Cellular Interactive Automata of the Rheology of Aetnean lava flows*, che in siciliano è il percorso della lava ormai solidificata) è un modello ad AC per la modellizzazione e la simulazione di colate laviche di tipo etneo. Il modello, illustrato brevemente in questa sezione, è stato testato con buoni risultati su numerose eruzioni etnee e si è dimostrato capace di seguire lo sviluppo di un evento e predirne l'evoluzione, di simulare gli effetti di possibili interventi orientati alla mitigazione del rischio (canali e/o ostacoli) ed è stato applicato alla creazione di mappe di rischio durante la crisi eruttiva del 2001 [17, 40, 38, 41, 56] e del 2002 [57].

Le assunzioni principali adottate nella modellizzazione dei flussi lavici nel modello SCIARA sono le seguenti:

- i flussi di lava tra le celle dell'AC sono determinati applicando l'algoritmo di minimizzazione delle differenze;
- la diminuzione della temperatura all'interno della cella è ottenuta considerando l'equazione d'irraggiamento;
- la diminuzione della temperatura determina la diminuzione del fattore di rallentamento dei flussi (calcolati tramite l'algoritmo di minimizzazione);
- l'adesione è definita come lo spessore di lava al disotto del quale tutti i flussi uscenti verso le vicine sono nulli; essa dipende dalla temperatura tramite una semplice relazione esponenziale inversa;
- l'effetto di solidificazione è banalmente ottenuto sommando lo spessore della lava nella cella alla quota e azzerando lo spessore di lava.

A ogni passo dell'AC è considerato un input dal mondo esterno che alimenta l'emissione di lava nelle celle dei crateri. La funzione di transizione, applicata a ogni cella dello spazio cellulare, determina l'evoluzione del sistema. I processi elementari più importanti sono qualitativamente (e brevemente) descritti di seguito.

Flussi di lava. I flussi uscenti dalle celle del reticolo verso le vicine sono calcolati tramite l'algoritmo di minimizzazione delle differenze. La resistenza reologica

della lava dipende in modo significativo dalla temperatura: più la temperatura è alta, più bassa è la resistenza e viceversa. Ai fini del comportamento dinamico della colata, la resistenza reologica determina l'immobilità di una parte di materiale lavico. Per tener conto di quest'effetto è stato introdotto un termine di adesione, adh, che rappresenta la quantità di lava, espressa in termini di spessore, che non può fuoriuscire dalla cella [123]. L'adesione varia con la temperatura secondo la seguente relazione: $adh = ae^{-bT}$ dove a e b sono costanti che dipendono dalla reologia della lava e T la temperatura [38]. L'algoritmo di minimizzazione è applicato, pertanto, alle seguenti quantità:

- $u(0) = q_a(0) + adh$
- $m = q_t(0) - adh$
- $u(i) = q_a(i) + q_t(i)$ ($i = 1, 2, \ldots, 4$ nel caso del reticolo quadrato con vicinato di von Neumann; $i = 1, 2, \ldots, 6$ nel caso di reticolo esagonale regolare)

essendo $q_a \in Q_a$ la quota e $q_t \in Q_t$ lo spessore di lava delle celle. Infine, il fattore di rallentamento varia da vicina a vicina e dipende della pendenza tra la cella centrale e la vicina considerata.

Variazioni di temperatura. Si assume che la quantità di lava nelle celle possa essere considerata omogenea dal punto di vista della distribuzione della temperatura. Di conseguenza, la stessa temperatura si ipotizza per i flussi uscenti verso le vicine. La variazione di temperatura è dovuta al movimento della lava che permette il mescolamento di flussi con differenti temperature (in realtà un mescolamento è permesso solo quando la differenza di temperatura non supera una certa soglia) e alla perdita d'energia termica dalla superficie. In un primo passo l'algoritmo considera i flussi entranti e uscenti e semplifica la situazione reale mediando la temperatura della lava residua e dei flussi entranti dai vicini. Un secondo passo stima la diminuzione di temperatura dovuta alla perdita d'energia termica alla superficie.

Solidificazione. Infine, se la temperatura scende al disotto di una soglia, l'effetto di solidificazione è banalmente ottenuto sommando lo spessore della lava nella cella alla quota e azzerando lo spessore di lava.

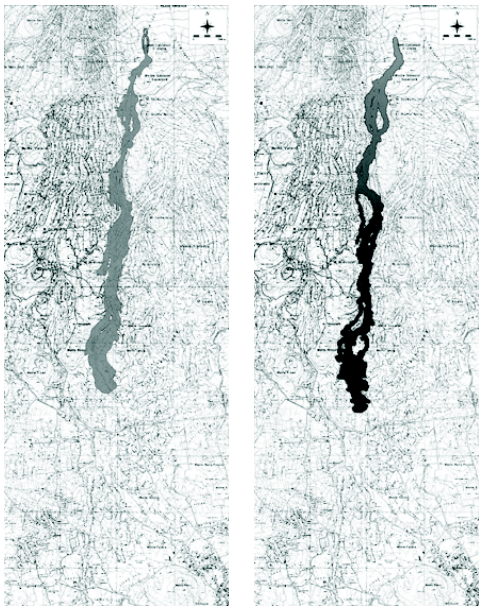


Figura 4.2: Confronto tra l'evento reale (a sinistra) relativo alla colata etnea del luglio-agosto 2001 e l'evento simulato (a destra) al decimo giorno con il modello SCIARA.

Il caso di studio della colata etnea del 2001

La validità del modello SCIARA è stata verificata tramite il confronto tra eventi simulati ed eventi reali sia in termini d'evoluzione spaziale che in termini di durata temporale [17, 56]. SCIARA è stato inoltre utilizzato nel corso dell'eruzione etnea del luglio-agosto 2001, evento della durata di dieci giorni che ha determinato una situazione di reale pericolo per i paesi di Belpasso e Nicolosi. La figura 4.2 si riferisce al confronto tra l'evento reale e l'evento simulato al decimo giorno. Successive simulazioni hanno permesso di realizzare scenari ipotetici per colate con differenti durate e intensità. La figura 4.3 mostra la simulazione di un evento ipotetico di 60 giorni con un tasso eruttivo di 12 m^3/s. Studi di questo tipo sono utili per la valutazione del rischio: in questo caso, per esempio, Nicolosi è risultato essere in posizione sicura dato che la colata ha formato un campo lavico (un accumulo di lava) prima della periferia del paese.

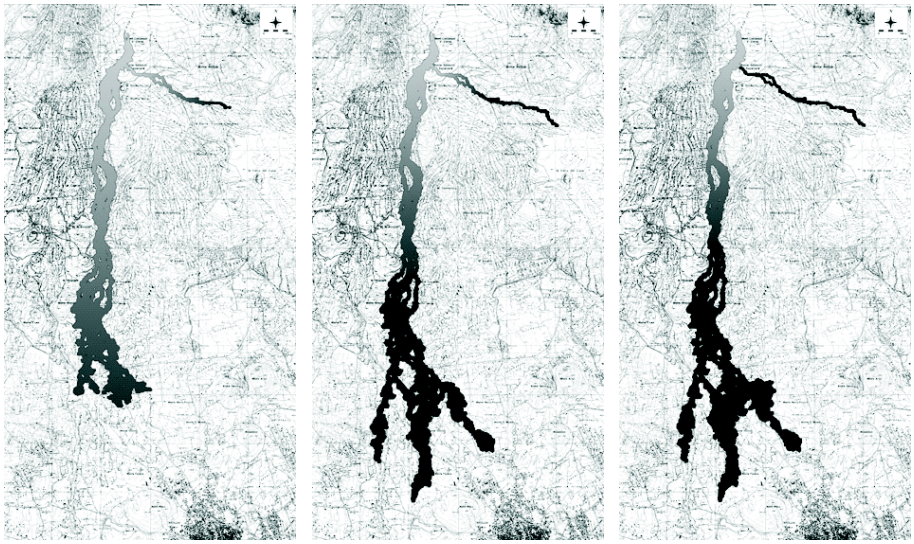


Figura 4.3: Ipotetico scenario di un evento lavico realizzato con il modello SCIARA su dati morfologici e reologici relativi all'evento del luglio-agosto 2001. La sequenza illustra rispettivamente l'evoluzione a 15, 45 e 60 giorni, considerando un tasso d'emissione di lava pari a 12 m^3/s. La posizione dei crateri coincide con l'effettiva posizione dei crateri reali.

4.3 Conversione dei dati geografici digitali in reticoli esagonali

Così come per i Gas Reticolari e per i modelli di Boltzmann su reticolo, è stato mostrato che l'adozione del reticolo esagonale può introdurre significativi miglioramenti nella simulazione di fenomeni macroscopici complessi [10, 38]. Tuttavia, alcune informazioni, solitamente necessarie alla simulazione (per esempio la topografia del suolo su cui il fenomeno evolve), sono disponibili sotto forma di reticoli a maglia quadrata e non esistono programmi specializzati nella gestione e manipolazione di dati geografici (GIS - Geographic Information Sistem) in grado di eseguire l'elaborazione delle informazioni in formato esagonale o, quanto meno, in grado di eseguire un qualche tipo di conversione in quel formato.

Come accennato in precedenza, le mappe topografiche digitali (DEM – Digital Elevation Model) sono rappresentate tramite griglie bidimensionali in cui ogni nodo (cella) contiene la quota media della regione di spazio che rappresenta. Quando una mappa digitale contiene informazioni su più caratteristiche del terreno, per esempio sulla quota, sull'erosione media, ecc., si parla di modelli digitali del terreno (DTM

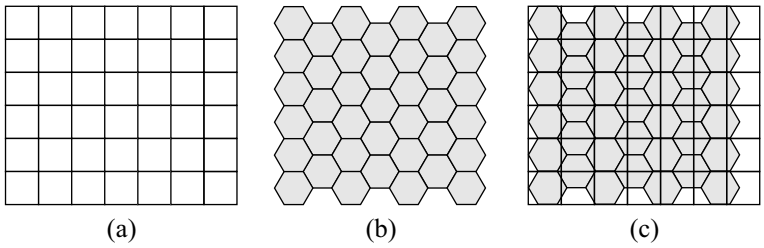


Figura 4.4: Procedura di conversione di un reticolo a maglie quadrate in un reticolo a maglie esagonali regolari. (a) rappresenta un generico reticolo a maglia quadrata, (b) un reticolo esagonale regolare con apotema dell'esagono pari a $l/2$, essendo l il lato del quadrato del primo reticolo; (c) gli effetti della sovrapposizione dei due reticoli. Il valore da assegnare alle celle del nuovo reticolo esagonale è calcolato come la media dei valori dei quadrati le cui porzioni dai area ricadono nell'esagono considerato, pesata sulle porzioni di area dei quadrati che l'esagono ricopre.

- Digital Terrain Model). In questo caso si tratta di un insieme di griglie a maglia quadrata, una per ogni caratteristica. I valori numerici delle grandezze considerate (quote, spessori di suolo eroso, ecc.) vengono rappresentati dal punto di vista matematico sotto forma di matrici, dal punto di vista informatico sotto forma di array bidimensionali.

Questo rende del tutto naturale la gestione, ed eventualmente la manipolazione, dei DTM da parte di AC bidimensionali con reticolo a maglia quadrata: basta infatti porre il parametro p_c pari alla distanza tra due punti successivi della griglia del DTM (dettaglio della mappa) per realizzare una corrispondenza biunivoca tra le celle dell'AC e i punti del modello digitale del terreno. In altri termini, per esempio nel caso della topografia, così facendo la mappa del sottostato quota dell'AC e il DEM della mappa digitale del terreno risultano perfettamente sovrapponibili.

Lo stesso discorso non vale, però, per AC con reticolo esagonale regolare. Una possibile soluzione consiste nell'"esagonalizzare" il DTM, cioè nell'operare una trasformazione che mappi i reticoli quadrati del DTM su griglie di celle esagonali regolari. Un metodo per l'esagonalizzazione di mappe digitali del terreno è stata proposta da D'Ambrosio et al. [48]. Il metodo mappa un generico reticolo a maglia quadrata in un reticolo esagonale regolare con apotema dell'esagono pari a $l/2$, essendo l il lato del quadrato (si veda la figura 4.4). Per un DEM il metodo consiste nell'assegnare a ogni esagono della nuova mappa la quota media dei quadrati parzialmente ricoperti dall'esagono considerato, pesata sulle rispettive porzioni areali. La figura 4.5 illustra un caso particolare d'applicazione del metodo

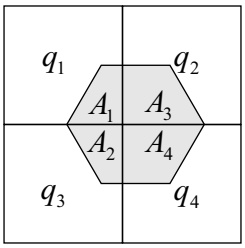


Figura 4.5: Caso particolare della procedura d'esagonalizzazione dei DTM: $q_1, \ldots, q_4$ rappresentano le quote delle celle del reticolo quadrato; $A_1, \ldots, A_4$ la misura delle aree che risultano dall'intersezione tra l'esagono e i quattro quadrati coinvolti nel processo. La quota dell'esagono, $q_{esagono}$, è calcolata tramite la seguente media pesata: $q_{esagono} = (\sum_{i=1}^{4} q_i A_i)/(\sum_{i=1}^{4} A_i)$.

d'esagonalizzazione. Lo stesso vale, ovviamente, per ogni altra informazione del DTM, come una mappa dell'erosione o una mappa della densità di vegetazione in una data area.

La sezione successiva presenta la famiglia di modelli SCIDDICA, sviluppati per la modellizzazione e simulazione di colate detritiche. La versione S3-hex, illustrata in dettaglio insieme alle applicazioni al disastro di Sarno del maggio 1998, è basata su un reticolo esagonale regolare. La fase preliminare di conversione dei dati geografici in formato esagonale è stata eseguita con il metodo sopra descritto.

4.4 Modellizzazione di flussi di detrito: il modello SCIDDICA

Il modello SCIDDICA (*Simulation through Computational Innovative methods for the Detection of Debris flow path using Interactive Cellular Automata*, che in siciliano significa scivola) è stato originariamente sviluppato per la simulazione di colate detritiche caratterizzate da moto puramente gravitazionale. Nelle successive versioni, tuttavia, è stato possibile gestire fenomeni di complessità più elevata aggiungendo progressivamente nuove interazioni locali e/o trasformazioni interne; SCIDDICA può essere considerato, per questo, un modello che si è evoluto in maniera incrementale [7, 8].

La versione "T" del modello è stata applicata alla modellizzazione e simulazione della frana di Tessina (Italia, 1992), caratterizzata da velocità estremamente bassa, dell'ordine dei metri al giorno, e moto puramente gravitazionale [9]. Nel modello

"T", l'algoritmo di minimizzazione è implementato nella sua forma più semplice. Inoltre, soltanto due processi elementari compongono la funzione di transizione (a) flussi uscenti (interazione locale) e (b) aggiornamento dello spessore di detrito nelle celle (interazione locale).

La versione "O", applicata alla frana del Monte Ontake (Giappone, 1984), è un'importante estensione del modello "T" [58]. Il caso di studio considerato è, infatti, caratterizzato da velocità decisamente più elevata (20-26 m/s) rispetto al caso di Tessina, con significativi effetti di risalita [161]. Si è reso necessario, di conseguenza, tener conto della capacità del flusso di avanzare in contropendenza e superare ostacoli morfologici; a tale scopo la funzione di transizione del modello si è arricchita del processo (c) determinazione del run-up (interazione locale) e l'algoritmo di minimizzazione è stato modificato in modo da tener conto degli effetti di risalita.

E' necessario sottolineare che l'accentuata irregolarità morfologica del Monte Ontake produce grande turbolenza e dissipazione d'energia. Di conseguenza, la strategia empirica basata sull'algoritmo di minimizzazione, opportunamente arricchita del processo elementare (c), rappresenta un'adeguata approssimazione in caso di flussi molto veloci [58]. Tuttavia, nel caso di situazioni morfologiche più regolari (per esempio una zona piana e/o con lieve pendenza), la dissipazione d'energia risulta minore e la direzione della quantità di moto è meno soggetta a variazioni improvvise (proprio per la mancanza di ostacoli morfologici). In tal caso gli effetti inerziali del flusso possono giocare un ruolo significativo nella dinamica del sistema e l'algoritmo di minimizzazione, così come proposto nella versione "O" del modello SCIDDICA, può risultare non completamente adeguato [49].

La successiva famiglia "Sx" del modello SCIDDICA è stata sviluppata per la simulazione delle colate detritiche di Sarno (Italia, 1998), caratterizzate da significativi effetti erosivi della copertura detritica lungo il percorso del flusso. D'Ambrosio et al. [51] hanno sviluppato la versione S1 introducendo il processo elementare della "mobilizzazione del suolo", originariamente specificato in due differenti fasi: 1) attivazione diretta per effetto del flusso sulla copertura detritica e 2) propagazione della mobilizzazione per "contatto" alle celle del vicinato. La funzione di transizione si è arricchita di due ulteriori processi elementari rispetto alla versione "O": (d) attivazione e propagazione della mobilizzazione del suolo erodibile (interazione locale in due fasi) ed (e) effetto della mobilizzazione (trasformazione interna) che trasforma il suolo eroso in detrito franante.

In SCIDDICA S2 D'Ambrosio et al. [50] hanno rivisto il processo elementare (a), responsabile della determinazione dei flussi uscenti, introducendo un più accu-

rato calcolo delle condizioni d'equilibrio nel vicinato. Di conseguenza, il processo elementare (a) è stato sostituito dal nuovo processo (a'). Inoltre, i processi elementari (b), aggiornamento dello spessore di detrito nelle celle per effetto dei flussi e (c), determinazione del run-up, sono stati riorganizzati nel seguente modo: (b'), aggiornamento del run-up e dello spessore di detrito (interazione locale) e (c') perdita di run-up per attrito (trasformazione interna).

D'Ambrosio et al. [49] hanno successivamente proposto il modello SCIDDICA S3-hex, descritto più avanti nei dettagli, introducendo un reticolo a maglia esagonale regolare in sostituzione dell'originario reticolo a maglia quadrata e migliorando il processo responsabile dell'erosione del suolo. Il nuovo modello è stato applicato da D'Ambrosio et al. alla simulazione delle frane di Sarno (Campania, 1998) [49] e alla simulazione del disastro di Cervinara e di San Martino Valle Caudina (Campania, 1999) [91]; un'applicazione della versione preliminare, basata ancora sul reticolo a maglia quadrata, alla simulazione delle frane di Sarno, è stata presentata da Iovine et al. [90].

Nella versione S3-hex, la doppia fase del processo (d), responsabile dell'attivazione e della propagazione della mobilizzazione, è stata unificata e considerata nel nuovo processo (d'), attivazione della mobilizzazione ed effetti, al quale è stato, inoltre, introdotto il meccanismo dell'"erosione progressiva" del manto detritico erodibile. Infine, per motivi computazionali, i processi elementari (b'), aggiornamento del run-up e dello spessore di detrito, e (c'), perdita di run-up per attrito, sono stati modificati in: (b"), aggiornamento dello spessore di detrito e dell'energia (interazione locale), e (c"), perdita d'energia per attrito (trasformazione interna).

4.4.1 Definizione formale del modello SCIDDICA S3-hex

Definizione 4.4.1. Il modello SCIDDICA S3-hex è formalmente definito nel seguente modo:

$$SCIDDICA\ S3-hex =< L, E, X, Q, P, \sigma, \gamma >$$

dove:

$L = \{(x, y) \in \mathbb{Z}^2 | -l_x < x < l_x,\ -l_y < y < l_y\}$ identifica lo spazio cellulare esagonale; $\mathbb{Z}$ è l'insieme dei numeri interi; l_x ed l_y identificano i limiti della regione in cui evolve il fenomeno;

$E \subset L$ è l'insieme delle celle in cui s'innesca il processo franoso (sorgenti);

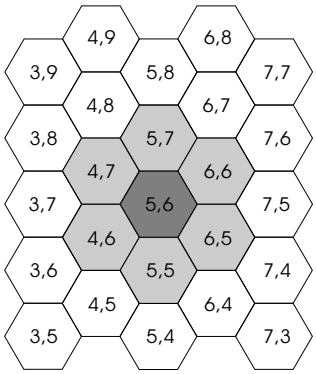


Figura 4.6: Spazio cellulare esagonale e vicinato. In grigio scuro la cella centrale di coordinate (5,6), in grigio chiaro le rimanenti vicine. Le coordinate delle celle del vicinato si ottengono sommando alle coordinate della cella centrale i vettori dell'insieme $X = \{(0,0),(1,0),(0,1),(0,-1),(-1,0),(1-,1),(1,-1)\}$ che definisce la relazione di vicinanza dell'AC.

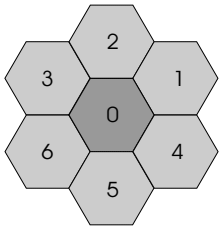


Figura 4.7: Vicinato esagonale del modello SCIDDICA S3-hex. La cella centrale è individuata dall'indice 0; gli indici da 1 a 6 individuano le rimanenti celle.

$X = \{(0,0),(1,0),(0,1),(0,-1),(-1,0),(1-,1),(1,-1)\}$ è la relazione di vicinanza (figura 4.6); le celle del vicinato sono indicizzate da 0 a 6, come illustrato in figura 4.7;

$Q = Q_a \times Q_{th} \times Q_e \times Q_d \times Q_o^6$ è l'insieme finito degli stati dell'ae, espresso come prodotto cartesiano dei sottostati considerati; $q_x \in Q_x$ rappresenta il valore q_x del sottostato Q_x:

- Q_a rappresenta la quota della cella;
- Q_{th} rappresenta lo spessore dei detrito di frana;
- Q_e rappresenta l'energia del detrito;
- Q_d rappresenta la profondità dello strato di suolo erodibile;

- Q_o^6 rappresentano i sei flussi di detrito uscenti dalla cella centrale verso le celle adiacenti, anch'essi in termini di spessore.

$P = \{p_a, p_t, p_{adh}, p_f, p_r, p_{rl}, p_{mt}, p_{er}\}$ è l'insieme dei parametri dell'AC:

- p_a rappresenta l'apotema della cella;
- p_t rappresenta il tempo corrispondente a un passo dell'AC;
- p_{adh} rappresenta lo spessore di detrito che non può essere distribuito alle celle adiacenti per effetto dell'adesione;
- p_f rappresenta l'angolo minimo tra la cella centrale e l'i-esima vicina perchè quest'ultima possa ricevere un flusso di detrito;
- p_r rappresenta il fattore di rallentamento dell'algoritmo di minimizzazione;
- p_{rl} rappresenta la perdita di run-up dovuta agli effetti d'attrito;
- p_{mt} rappresenta la soglia d'energia necessaria a innescare il processo di mobilizzazione della copertura detritica erodibile;
- p_{er} rappresenta il fattore di erosione progressiva;

$\sigma : Q^7 \rightarrow Q$ è la funzione di transizione deterministica dell'ae, costituita dai seguenti processi elementari, applicati nello stesso ordine in cui sono presentati:

1. trasformazione interna T_1: attivazione ed effetto della mobilizzazione (d');
2. interazione locale I_1: flussi uscenti (a');
3. interazione locale I_2: aggiornamento dello spessore di detrito e dell'energia (b");
4. trasformazione interna T_2: perdita d'energia (c").

$\gamma : E \times \mathbb{N} \times Q_d \times Q_{th} \times Q_e \rightarrow Q_a \times Q_d \times Q_{th} \times Q_e$ è la funzione d'attivazione delle sorgenti, responsabile dell'innesco degli eventi franosi a passi prefissati dell'AC. Le sorgenti principali s'innescano al primo passo; ulteriori sorgenti possono innescarsi successivamente. $\mathbb{N}$ è l'insieme dei numeri naturali, corrispondenti ai passi dell'AC.

All'inizio di ogni simulazione gli stati delle celle devono specificare le condizioni iniziali del sistema. I valori iniziali sono assegnati nel seguente modo:

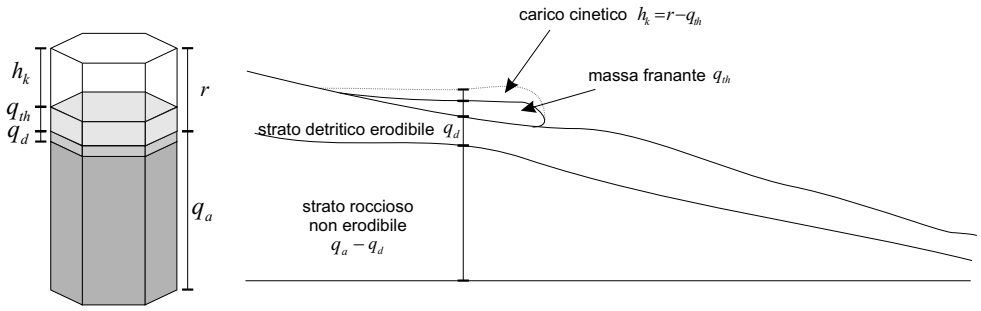


Figura 4.8: Modello del suolo in SCIDDICA S3-hex.

- q_a è posto uguale alla quota (quota topografica dello strato roccioso più spessore dello strato erodibile – si veda la figura 4.8);
- q_{th} è zero ovunque tranne che nelle aree di innesco della frana, dove vale q_d;
- q_e è zero ovunque tranne che nelle aree di innesco della frana dove è uguale all'energia del detrito;
- q_d è lo spessore di suolo erodibile (nelle celle d'innesco $q_d = 0$);
- q_o è zero ovunque.

La funzione di transizione è applicata alle celle dell'AC in modo che la configurazione del sistema cambi, e l'evoluzione della simulazione sia ottenuta.

4.4.2 Considerazioni generali

In idrodinamica [120, 158] il carico cinetico è definito come

$$h_k = v^2/2g$$

essendo v la velocità del flusso e g l'accelerazione di gravità. In accordo, il run-up r, cioè l'altezza che può essere raggiunta dal flusso, è definito come:

$$r = h + v^2/2g = h + h_k$$

essendo h l'altezza del flusso (figura 4.9).

Si consideri una colonna di base A, massa m e altezza h sul piano $z = 0$ (figura 4.9); la sua energia potenziale è:

$$U = \rho g A \int_0^h z dz = \left[\frac{z^2}{2}\right]_0^h = \frac{\rho g A}{2} h^2 \tag{4.4}$$

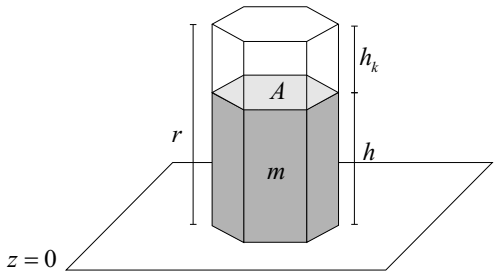


Figura 4.9: Esemplificazione dell'energia potenziale nel modello SCIDDICA S3-hex. La figura mostra una colonna di detrito sul piano $z = 0$: A è la base della colonna, h l'altezza, m la massa, h_k il carico cinetico ed r il run-up.

essendo ρ la densità del materiale che costituisce la colonna. L'effetto del carico cinetico può essere inserito nell'equazione 4.4 se si incrementa virtualmente l'altezza della colonna da h ad r. Poichè la massa deve essere conservata, è necessario considerare una nuova densità ρ':

$$\rho' = \frac{h}{r}\rho < \rho$$

La seguente formula esprime il conseguente aumento d'energia:

$$U' = \frac{\rho' g A}{2} r^2 = \frac{h}{r}\rho \frac{gA}{2} r^2 = \frac{\rho g A}{2} hr > U$$

Nel modello S3-hex, q_e rappresenta l'energia U' (riferita alla quota della cella), q_{th} rappresenta lo spessore di detrito h. Di conseguenza:

$$q_e = \frac{\rho g A}{2} q_{th} r$$

Il termine $\rho gA/2$ può essere considerato costante: infatti A è l'area della cella, e ρ può essere assunto costante nelle applicazioni del modello. Di conseguenza r è proporzionale a q_e/q_{th} secondo il fattore $k = 2/\rho gA$:

$$r = \frac{2q_e}{\rho g A q_{th}} = k\frac{q_e}{q_{th}} \tag{4.5}$$

La figura 4.10 mostra un esempio di un flusso da una cella verso una vicina, quest'ultima caratterizzata da un'altezza maggiore. Per una dettagliata discussione del meccanismo di distribuzione dei flussi si veda la sezione successiva.

La figura 4.8 mostra il carico cinetico h_k, il run-up r, la quota della cella $q_a \in Q_a$ e lo spessore di detrito $q_{th} \in Q_{th}$ in confronto al modello di suolo e alle assunzioni idrodinamiche considerate nel modello SCIDDICA S3-hex.

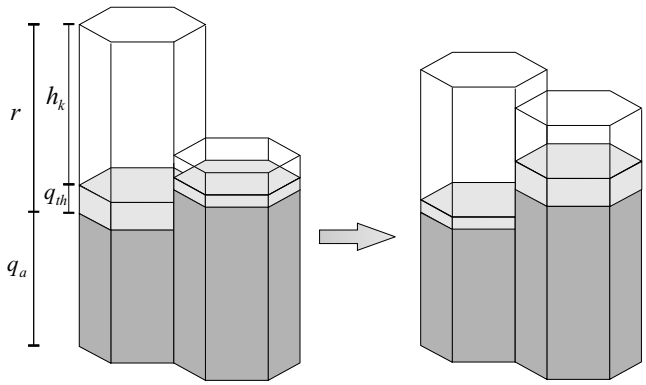


Figura 4.10: Esempio di distribuzione del detrito nel modello SCIDDICA S3-hex in cui la cella che riceve detrito è caratterizzata da un'altezza (in termini di quota più lo spessore di detrito) maggiore rispetto a quella che ne cede. A sinistra la situazione al pass t, a destra la situazione al passo $t + 1$.

4.4.3 La funzione di transizione del modello SCIDDICA S3-hex

La sezione illustra nei dettagli i processi elementari che definiscono la funzione di transizione del modello SCIDDICA S3-hex.

Interazione locale I_1: determinazione dei flussi uscenti di detrito

L'interazione locale I_1, definita dalla funzione

$$\sigma_{I_1} : Q_a^7 \times Q_{th}^7 \times Q_e \rightarrow Q_o^6$$

determina i flussi di detrito uscenti dalla cella centrale verso le celle adiacenti. Essa si basa su un opportuno algoritmo di minimizzazione delle differenze, derivato dall'originario algoritmo proposto da Di Gregorio e Serra [59].

Allo scopo di considerare gli effetti del run-up, l'altezza della colonna di detrito nella cella centrale è virtualmente incrementata da $h = q_{th}(0)$ a $r = kq_e/q_{th}$ (equazione 4.5) e l'algoritmo di minimizzazione è applicato alle seguenti quantità:

- $u(0) = q_a(0) + p_{adh}$;
- $m = r - p_{adh}$;
- $u(i) = q_a(i) + q_{th}(i)$ $(i = 1, 2, \ldots, 6)$;

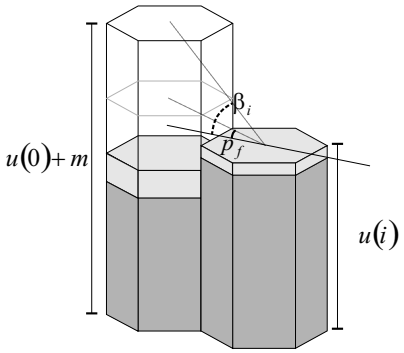


Figura 4.11: Rappresentazione dell'angolo di frizione. β_i rappresenta l'angolo tra la cella centrale (cella a sinistra) e l'i-esima vicina (cella a destra); Se $\beta_i < p_f$, la vicina i-esima viene eliminata e non può ricevere flusso dalla cella centrale.

Precedentemente all'esecuzione dell'algoritmo di minimizzazione è eseguito un test preliminare: è calcolato l'angolo β_i tra la cella centrale, in termini d'altezza $u(0) + m$, e l'i-esima vicina, anch'essa in termini d'altezza $u(i)$, (si veda la figura 4.11) e le celle per le quali risulti $\beta_i < p_f$ sono eliminate dal calcolo dalla media e dalla distribuzione dei flussi.

L'algoritmo di minimizzazione è eseguito solo nel caso in cui risulti $q_{th}(0) - p_{adh} > 0$, essendo quest'ultima quantità quella che verrà effettivamente distribuita alle celle adiacenti. L'applicazione dell'algoritmo determina i flussi "virtuali" $f(i)$ $(i = 1, 2, \ldots, 6)$, essendo in questa fase considerata come quantità distribuibile il run-up anzichè il detrito; gli effettivi flussi di detrito, $q_o(0, i)$ $(i = 1, 2, \ldots, 6)$, sono ottenuti moltiplicando gli $f(i)$ per il fattore di normalizzazione $v_{nf} = h/r$ e per il fattore di rallentamento p_r:

$$q_o(0, i) = v_{nf} f(i) p_r$$

La figura 4.12 mostra un esempio d'applicazione dell'algoritmo di minimizzazione utilizzato nel modello SCIDDICA S3-hex.

Interazione locale I_2: aggiornamento dello spessore di detrito e dell'energia

L'interazione locale I_2, definita dalla funzione

$$\sigma_{I_2} : (Q_{th} \times Q_e \times Q_o^6)^7 \rightarrow Q_{th} \times Q_e$$

aggiorna i valori di q_{th} e q_e dei sottostati spessore di detrito ed energia.

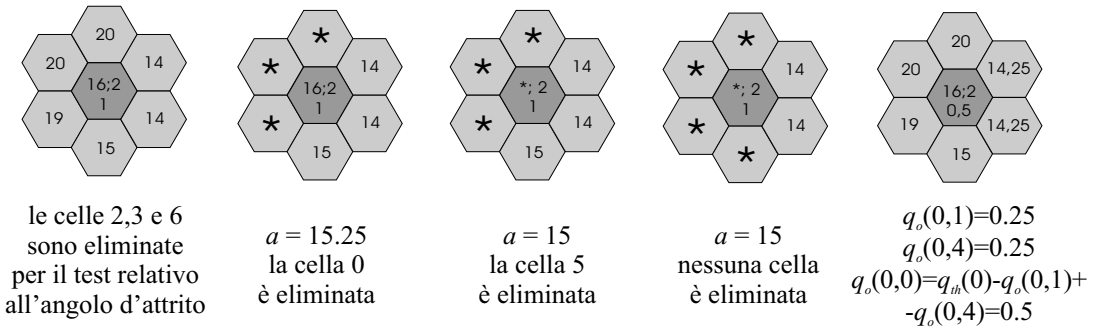


Figura 4.12: Esempio d'applicazione dell'algoritmo di minimizzazione nel caso del modello SCIDDICA S3-hex. L'indicizzazione delle celle del vicinato esagonale è illustrata in figura 4.7. Il simbolo * indica l'eliminazione della cella dal calcolo della media e dalla distribuzione del flusso. Per semplicità si considera $p_{adh} = 0$ e $p_r = 1$. Nella cella centrale sono considerate due quantità, una inamovibile, $u(0) = q_a(0) + p_{adh} = q_a(0) = 16$ e una movibile $m = r - p_{adh} = r = 2$. Inoltre la quantità effettivamente distribuibile è $q_{th} = 1$, da cui si ricava il fattore di normalizzazione $v_{nf} = q_{th}/r = 0.5$. Quando nessuna cella è eliminata, sono calcolati i flussi preliminari $f(1) = f(4) = 0.5$; i flussi effettivi di detrito sono ottenuti moltiplicando questi ultimi per il fattore di normalizzazioe e per il fattore di rallentamento: $q_o(0,1) = q_o(0,4) = v_{nf} f(4) p_r = 0.5 \cdot 0.5 \cdot 1 = 0.25$; i rimanenti flussi sono nulli. Il detrito residuo nella cella centrale è calcolato sottraendo a $q_{th}(0)$ i flussi uscenti non nulli: $q_o(0,0) = q_{th}(0) - q_o(0,1) - q_o(0,4) = 1 - 0.25 - 0.25 = 0.5$.

Il nuovo valore dello spessore di detrito nella cella, $new\ q_{th}$, è ottenuto considerando le variazioni dello spessore di detrito dovute ai flussi entranti e uscenti:

$$new\ q_{th} = q_{th}(0) + \sum_{i=1}^{6}(q_o(i,0) - q_o(0,i))$$

Come prima, il nuovo valore dell'energia, $new\ q_e$, è ottenuto considerando le variazioni dovute ai flussi entranti e uscenti:

$$new\ q_e = \left(q_{th}(0) - \sum_{i=1}^{6} q_o(0,i)\right) \cdot \left(k\frac{q_e(0)}{q_{th}(0)}\right) + \sum_{i=1}^{6}\left(q_o(i,0) \cdot \left(k\frac{q_e(i)}{q_{th}(i)}\right)\right)$$

Trasformazione interna T_1: attivazione ed effetto della mobilizzazione

La trasformazione interna T_1, definita dalla funzione

$$\sigma_{T_1} : Q_a \times Q_e \times Q_{th} \times Q_d \rightarrow Q_a \times Q_e \times Q_{th} \times Q_d$$

determina l'erosione del suolo e i suoi effetti.

La condizione d'erosione è: $q_e(0) > p_{mt}$; la quantita di suolo eroso è: $\Delta_d = (q_e(0) - p_{mt})p_{er}$ se $\Delta_d < q_d(0)$, altrimenti $\Delta_d = q_d(0)$. I seguenti nuovi valori aggiornano rispettivamente i sottostati quota, profondità della copertura detritica erodibile, spessore di detrito e run-up:

$$\begin{aligned} new\ q_a &= q_a(0) - \Delta_d \\ new\ q_d &= q_d(0) - \Delta_d \\ new\ q_{th} &= q_{th}(0) + \Delta_d \\ new\ r &= r + \Delta_d \end{aligned}$$

L'energia è calcolata in riferimento allo spessore di detrito e al run-up ed è aggiornata in accordo alla seguente formula:

$$new\ q_e = \frac{1}{k}(q_{th}(0) + \Delta_d)(r + \Delta_d) = \frac{1}{k}(q_{th}(0) + \Delta_d)\left(k\frac{q_e(0)}{q_{th}(0)} + \Delta_d\right)$$

Trasformazione interna T_2: perdita d'energia

La trasformazione interna T_2, definita dalla funzione

$$\sigma_{T_2} : Q_e \times Q_{th} \rightarrow Q_e$$

determina la perdita d'energia per attrito. Quest'ultima è ottenuta riducendo il run-up a un valore, non inferiore a $q_{th}(0)$, attraverso il parametro p_{rl}. La perdita di run-up è $\Delta_r = p_{rl}$, se $(kq_e(0)/q_{th}(0) - p_{rl}) > q_{th}(0)$, altrimenti $\Delta_r = kq_e(0)/q_{th}(0) - q_{th}(0)$. Questo implica che

$$new\ q_e = q_e(0) - \frac{1}{k}\Delta_r q_{th}(0)$$

4.4.4 La funzione d'attivazione delle sorgenti d'innesco γ

La funzione d'attivazione delle sorgenti è definita nel seguente modo:

$$\gamma : E \times \mathbb{N} \times Q_d \times Q_{th} \times Q_e \rightarrow Q_a \times Q_d \times Q_{th} \times Q_e$$

essendo $E \subset L$ l'insieme delle celle che specificano i punti d'innesco ed $\mathbb{N}$ l'insieme dei numeri naturali, corrispondenti ai passi dell'AC.

Le sorgenti principali s'innescano al primo passo dell'AC, mentre ulteriori sorgenti secondarie possono innescarsi successivamente, a passi prefissati dell'AC. Quando una sorgente s'innesca sono considerate le seguenti variazioni:

$$\begin{aligned} new\ q_a &= q_a(0) - q_d(0) \\ new\ q_d &= q_d(0) - q_d(0) = 0 \\ new\ q_{th} &= q_{th}(0) + q_d(0) \\ new\ r &= r + q_d(0) \end{aligned}$$

L'energia è ricalcolata in riferimento allo spessore di detrito e al run-up ed è aggiornata in accordo alla seguente formula:

$$new\ q_e = \frac{1}{k}(q_{th}(0) + q_d(0))(r + q_d(0)) = \frac{1}{k}(q_{th}(0) + q_d(0))\left(k\frac{q_e(0)}{q_{th}(0)} + q_d(0)\right)$$

4.5 Applicazioni del modello SCIDDICA S3-hex

4.5.1 L'evento del maggio 1998 nell'area di studio di Pizzo d'Alvano

Nel corso degli ultimi decenni, numerosi studi hanno tentato di caratterizzare [101, 4, 184] e di modellizzare [121, 5, 92] le colate detritiche. Note nel gergo tecnico con

il termine generico di debris flow [99], le colate detritiche sono misture di acqua e sedimenti che si muovono, anche a grandi velocità, come fluidi viscosi, generalmente incanalandosi lungo gli impluvi minori e i corsi dei torrenti. I casi più distruttivi sono rappresentati da misture eterogenee dal punto di vista granulometrico, con pezzature variabili dalle argille ai blocchi di diversi metri cubi.

I debris flow si originano, in genere, durante temporali molto intensi, a seguito di precipitazioni prolungate, o per rapida fusione nivale. La mobilizzazione del materiale detritico può anche avvenire a seguito di sollecitazioni dinamiche, come quelle causate da impatti, vibrazioni, scosse sismiche, oppure in concomitanza con eruzioni vulcaniche [134]. In particolare, il materiale che viene coinvolto in un fenomeno di flusso detritico si trova, prima dell'evento, in condizioni di equilibrio (sebbene precario) su di un versante o lungo un impluvio: è necessario, pertanto, l'intervento di forze destabilizzanti che distruggano innanzitutto lo scheletro del materiale in posto e, successivamente, sostengano le particelle disperse nel fluido. La condizione necessaria per la distruzione dello scheletro del suolo è comune al caso di una frana generica. Tuttavia, per far sì che il materiale, una volta collassato, continui a propagarsi verso valle con caratteri di colata, sono indispensabili anche altri due fattori, cioè: valori notevoli di pendenza (fonte energetica, connessa ai caratteri morfologici) e disponibilità d'acqua (per riempire gli spazi venutisi a creare tra le particelle) [89].

Nella maggior parte dei casi, i debris flow hanno origine come scivolamenti superficiali di terreno (soil slip) [102]. Tuttavia, essi possono anche generarsi per mobilizzazione di sedimenti precedentemente accumulati lungo il reticolo di drenaggio (per rapida erosione lungo i canaloni) o per collasso di sbarramenti (naturali o artificiali) [63, 170].

Lo sviluppo dei debris flow è, in genere, rapido e non sempre preceduto da pròdromi apprezzabili: per tale motivo, essi hanno spesso colto di sorpresa le popolazioni, preoccupate, in quei frangenti, più per le intense piogge o per l'eventualità di esondazioni fluviali. Tali frane possono raggiungere velocità anche molto elevate (fino a diecine di metri al secondo). In corrispondenza della fine del percorso confinato ed acclive, i debris flow decelerano rapidamente e distribuiscono i detriti nella zona di conoide posta allo sbocco del bacino.

Il 5 e 6 maggio 1998, principalmente sul versante del massiccio di Pizzo d'Alvano (Campania), si sono innescati migliaia di scivolamenti detritici e flussi detritici [55]. Lo scorrimento del materiale ha eroso in profondità lo strato detritico (variabile da pochi centimentri a qualche metro) lungo il cammino verso fondo valle e l'impatto con le aree urbanizzate ha prodotto ingenti danni e, purtroppo, 161 vittime.

Le figure 4.13a, 4.14a e 4.15a illustrano alcune delle principali frane verificatesi sul versante meridionale di Pizzo d'Alvano. Il flusso detritico di Chiappe di Sarno (figura 4.13a) inizialmente si propaga lungo un versante piano-convesso, per poi suddividersi in due tronconi; questi si ricongiungono in una fase successiva alla base del massiccio causando gravi danni nell'area di Curti. Il caso di studio di Curti (figura 4.14a) si innesca non molto lontano dall'evento di Chiappe di Sarno. Il fenomeno si propaga rapidamente verso valle incanalandosi nel reticolo idrografico principale (che erode in profondità) innescando, inoltre, quattro frane secondarie. Alla base del massiccio, il flusso si suffivide in due parti, ricongiungendosi in prossimità di Curti e causando due vittime. La frana di Pestello Storto (figura 4.15a) è un esempio di frana ben incanalata. Tuttavia, a fondo valle, il flusso impatta contro un muro che ne limita parzialemte l'avanzata.

I casi di studio appena descritti sono stati utilizzati per la calibratura del modello SCIDDICA S3-hex.

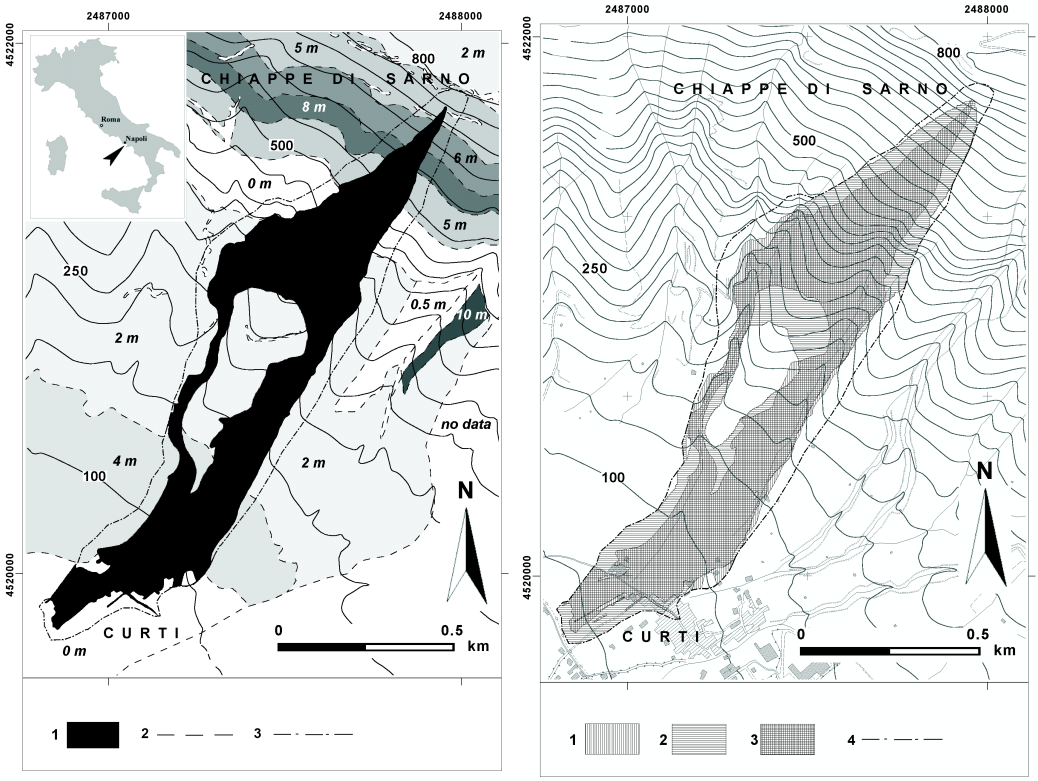
4.5.2 Applicazioni del modello ai casi di Chiappe di Sarno, Curti e Pestello Storto

I parametri del modello sono stati stimati considerando il confronto areale tra gli eventi reali e i migliori eventi simulati. Una fase di calibrazione preliminare è consistita nell'assegnare valori "ragionevoli" ai parametri del modello sulla base di precedenti simulazioni relative alla stessa area di studio, eventualmente eseguite anche con versioni precedenti del modello. I risultati sono stati analizzati in ambiente GIS comparando le aree coinvolte da entrambi gli eventi, reale e simulato, e quelle interessate da un solo evento per volta. Inoltre, anche il confronto degli effetti dell'erosione e della deposizione sono stati tenuti in considerazione nella valutazione dei risultati. Sulla base di tali analisi, i valori dei parametri del modello sono stati iterativamente modificati così da ottenere risultati sempre migliori.

Le simulazioni sono state sistematicamente confrontate con i casi reali e una stima quantitativa per la valutazione dei risultati si è basata sui seguenti indicatori:

$$e_1 = \sqrt{\frac{\mu(R \cap S)}{\mu(R \cup S)}}$$

$$e_2 = \sqrt{\frac{\mu(R \cap S)}{\mu(R)}}$$



(a) (b)

Figura 4.13: (a) La frana di Chiappe di Sarno: (1) identifica l'area affetta dalla frana; (2) identifica il limite delle zone con spessore (in corsivo) dello strato erodibile costante; (3) identifica il bordo dell'analisi eseguita in ambiente GIS. (b) Confronto tra frana reale e simulata: (1) identifica le aree affette dalla frana reale; (2) identifica le aree affette dalla frana simulata; (3) identifica le aree affette da entrambe le frane; (4) identifica il bordo dell'analisi eseguita in ambiente GIS.

essendo R l'area affetta dalla frana reale, S l'area affetta dalla frana simulata e $\mu : A \rightarrow [0, +\infty)$ la misura (in metri quadrati) dell'insieme A. Il valore di entrambi gli indicatori varia tra 0, che indica il completo fallimento della simulazione, e 1, che corrisponde alla simulazione perfetta. In particolare, quando $e_1 = 1$ vuol dire che le due frane (reale e simulata) si sovrappongono perfettamente dal punto di vista areale, mentre $e_2 = 1$ indica che la frana simulata ricopre completamente la reale. Nella valutazione quantitativa dei risultati è stato adottato il seguente criterio: le simulazioni sono state ritenute "accettabili" solo per valori degli indicatori sufficientemente alti: almeno 0.7 per l'indicatore e_1 e 0.85 per per l'indicatore e_2.

I migliori valori ottenuti per i parametri del modello SCIDDICA S3-hex sono i seguenti: $p_{adh} = 0.001\,m$; $p_f = 0.1\,m$; $p_r = 1$; $p_{rl} = 0.6\,m$; $p_{mt} = 3.5\,m^2$; $p_{er} = 0.015$. I parametri p_a e p_t non sono stati oggetto di calibratura: il parametro $p_a = 1.25\,m$ è stato settato sulla base del dettaglio del DTM dell'area di studio; il parametro $p_t \approx 0.01\,s$ è stato ricavato dall'osservazione *a posteriori* del comportamento del sistema (in termini di velocità media ed estensione del flusso) sui casi di studio considerati.

Si noti che il modello può essere effettivamente applicato solo a monte dei settori urbanizzati, cioè dove i processi di erosione del suolo possono essere propriamente modellati; inoltre in tali aree la precisione dei dati morfologici è sensibilmente migliore rispetto all'urbanizzato. Di conseguenza, i valori dei due indicatori, e_1 ed e_2, sono stati stimati nelle aree delimitate dalle linee tratteggiate illustrate nelle figure 4.13, 4.14 e 4.15.

La stima dei parametri dell'AC ha permesso di simulare con buona approssimazione i casi di studio considerati. Le figure 4.13b, 4.14b e 4.15b illustrano i risultati delle migliori simulazioni e il confronto con gli eventi reali. Dal punto di vista qualitativo, le caratteristiche essenziali del fenomeno sono state riprodotte; dal punto di vista quantitativo, i valori dei due indicatori (tabella 4.1) soddisfano i criteri imposti precedentemente. In particolare: a) il caso di Chiappe di Sarno ha evidenziato i risultati migliori per entrambi gli indicatori; b) i valori che caratterizzano i due indicatori per il caso di Curti sono anch'essi accettabili; c) per il caso di Pestello storto, il valore dell'indicatore e_2 è abbastanza buono, mentre il valore dell'indicatore e_1 è appena accettabile. Quest'ultimo risultato, tuttavia, potrebbe essere dovuto alla scarsa qualità dei dati topografici, specialmente nella parte alla base del massiccio.

La figura 4.16 illustra, infine, i risultati di una simulazione eseguita sull'intero versante meridionale del massiccio di Pizzo d'Alvano, adottando i valori dei parametri stimati nella fase di calibratura del modello.

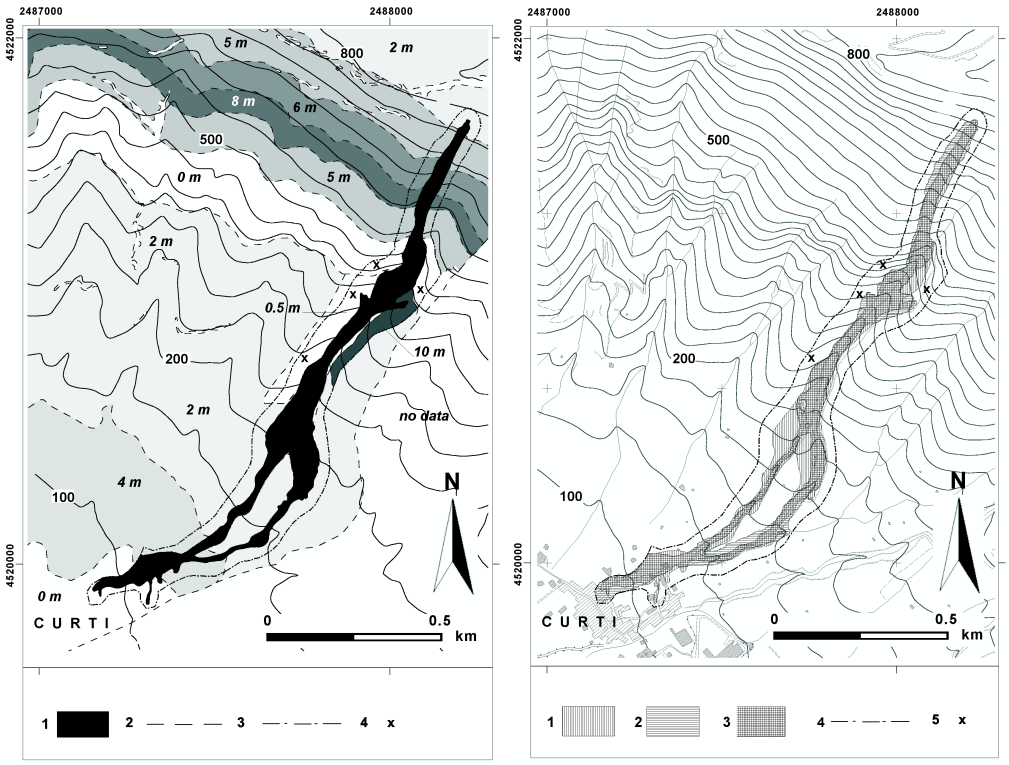
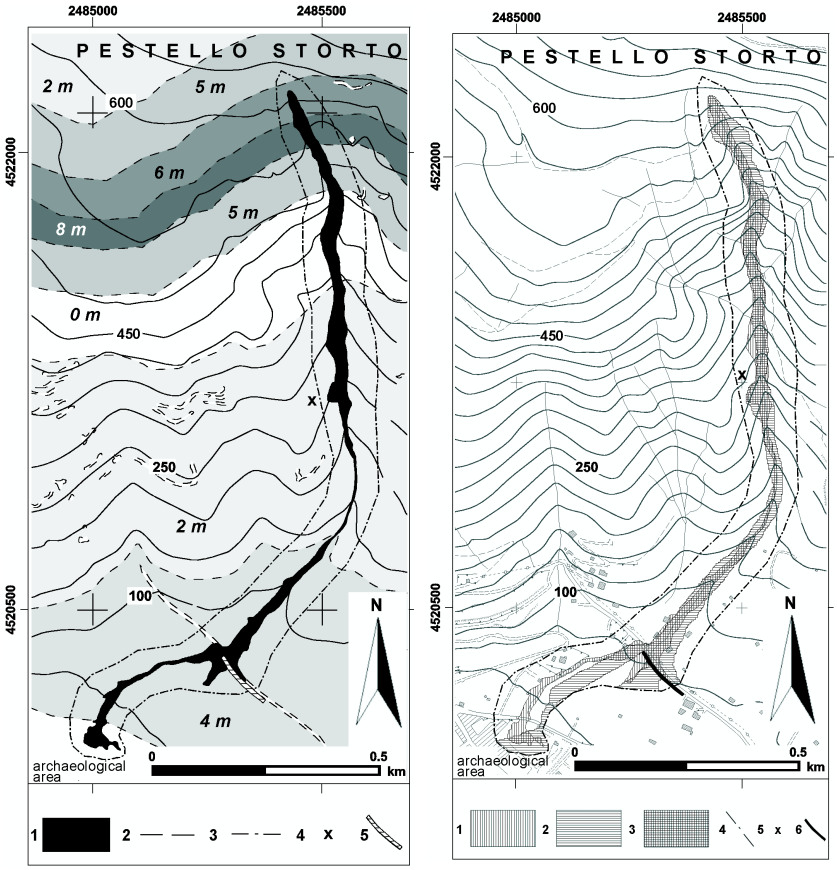


Figura 4.14: (a) La frana di Curti: (1) identifica l'area affetta dalla frana; (2) identifica il limite delle zone con spessore (in corsivo) dello strato erodibile costante; (3) identifica il bordo dell'analisi eseguita in ambiente GIS; (4) identifica i siti d'innesco secondarie. (b) Confronto tra frana reale e simulata: (1) identifica le aree affette dalla frana reale; (2) identifica le aree affette dalla frana simulata; (3) identifica le aree affette da entrambe le frane; (4) identifica il bordo dell'analisi eseguita in ambiente GIS; (5) identifica i siti d'innesco secondarie.

Caso	$R\,(m^2)$	$S\,(m^2)$	$R \cap S\,(m^2)$	$R \cup S\,(m^2)$	e_1	e_2
Chiappe di Sarno	319390	425660	300626	444423	0.82	0.97
Curti	104990	95334	75498	124826	0.78	0.85
Pestello Storto	40792	64116	30793	74114	0.64	0.87

Tabella 4.1: Risultati quantitativi dell'applicazione del modello SCIDDICA S3-hex ai casi di studio del versante meridionale di Pizzo d'Alvano.



(a) (b)

Figura 4.15: (a) La frana di Pestello Storto: (1) identifica l'area affetta dalla frana; (2) identifica il limite delle zone con spessore (in corsivo) dello strato erodibile costante; (3) identifica il bordo dell'analisi eseguita in ambiente GIS; (4) identifica i siti d'innesco secondarie; (5) identifica barriere naturali o artificiali. (b) Confronto tra frana reale e simulata: (1) identifica le aree affette dalla frana reale; (2) identifica le aree affette dalla frana simulata; (3) identifica le aree affette da entrambe le frane; (4) identifica il bordo dell'analisi eseguita in ambiente GIS; (5) identifica i siti d'innesco secondarie; (6) identifica barriere naturali o artificiali.

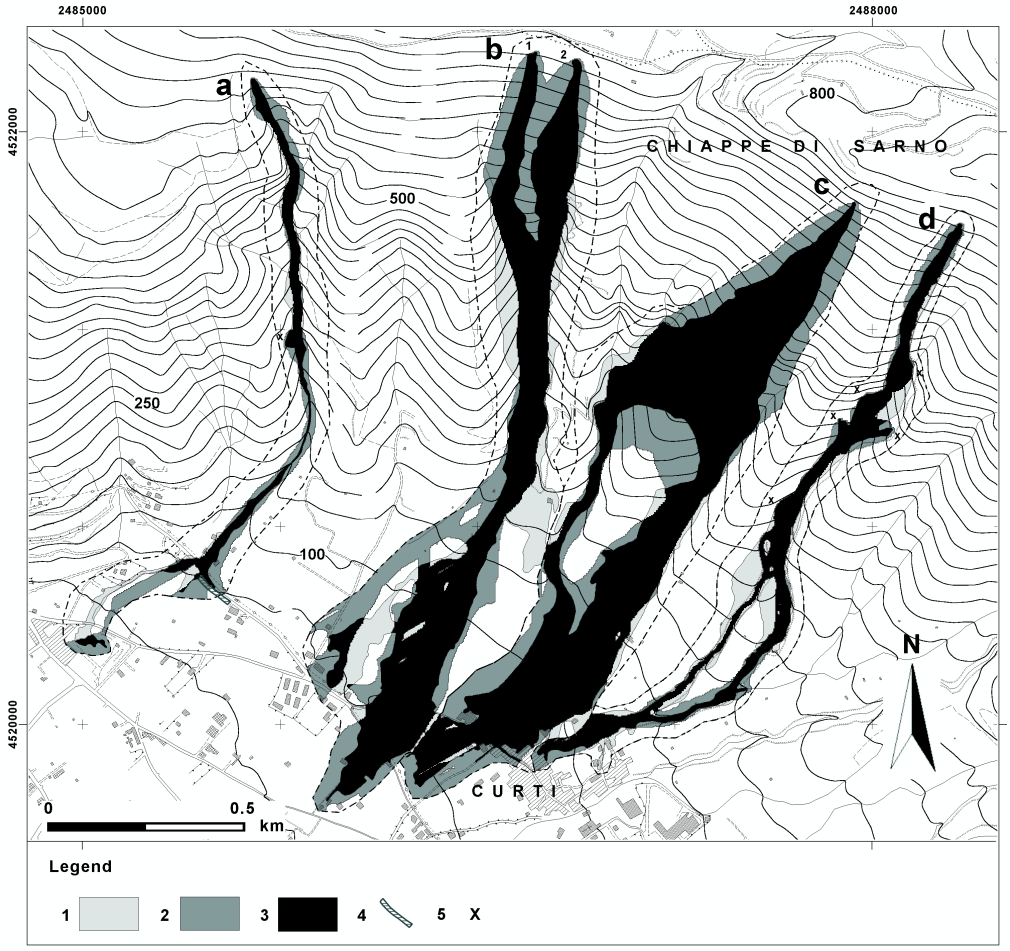


Figura 4.16: Simulazione della catastrofe del versante meridionale del massiccio di Pizzo d'Alvano: (1) identifica le aree affette dalla frana reale; (2) identifica le aree affette dalla frana simulata; (3) identifica le aree affette da entrambe le frane; (4) identifica il bordo dell'analisi eseguita in ambiente GIS; (5) identifica i siti d'innesco secondarie.

Capitolo 5

Effetti inerziali nella simulazione di flussi detritici e ottimizzazione con Algoritmi Genetici Paralleli

5.1 Introduzione

Come discusso nel capitolo precedente, gli Automi Cellulari possono rappresentare un valido strumento nella modellizzazione e simulazione di fenomeni complessi che evolvono sulla base di interazioni locali. Nel contesto del metodo empirico, proposto da Di Gregorio e Serra [59] per la modellizzazione di fenomeni macroscopici complessi, è stata sviluppata la famiglia dei modelli SCIDDICA per la simulazione di colate detritiche caratterizzate da differenti livelli di complessità: il modello "T" è stato proposto per la simulazione di frane caratterizzate da basse velocità ed è stato applicato al caso di studio di Tessina; il modello "O" è stato proposto per la simulazione di colate detritiche rapide e applicato al caso di studio del Monte Ontake; le versioni Sx sono state sviluppate, infine, per la simulazione di colate rapide caratterizzate da forti processi erosivi lungo il percorso del flusso. Le versioni Sx sono state tutte applicate alla simulazione delle frane di Sarno.

Nonostante i buoni risultati ottenuti (in particolar modo con la versione S3-hex grazie anche all'adozione del reticolo esagonale in sostituzione dell'originario reticolo a maglia quadrata) è stato compiuto un ulteriore sforzo nel tentativo di modellizzare le proprietà inerziali, tipiche degli eventi di Sarno, non considerate nei modelli precedenti. Iovine, D'Ambrosio e Di Gregorio hanno sviluppato il modello SCIDDICA S4a [88], mentre D'Ambrosio, Spataro e Iovine hanno proposto

il modello SCIDDICA S4b [52]: essi rappresentano i primi tentativi di introdurre, nel contesto specifico del metodo empirico considerato, gli effetti della quantità di moto.

L'introduzione degli effetti inerziali ha posto, tuttavia, pesanti problemi nella calibratura dei due modelli, ovvero nella ricerca dei valori dei parametri necessari affinchè le simulazioni riproducessero il più fedelmente possibile il comportamento degli eventi reali. Gli Algoritmi Genetici, discussi nel secondo capitolo, sono algoritmi di ricerca che si ispirano ai meccanismi della selezione naturale e della riproduzione sessuale. Essi hanno mostrato la loro efficacia nella risoluzione di un gran numero di problemi, e risultano particolarmente utili in quei settori applicativi in cui non esistono metodi d'ottimizzazione standard.

L'applicazione degli Algoritmi Genetici all'ottimizzazione dei modelli S4a ed S4b sul caso di studio di Curti ha permesso di determinare insiemi di valori dei parametri che, adottati nei due modelli, hanno consentito di riprodurre in maniera soddisfacente l'evento reale. Tuttavia, la fase di ottimizzazione del modello S4a, in cui gli Algoritmi Genetici sono stati applicati in un ambiente di calcolo sequenziale, ha richiesto tempi di calcolo dell'ordine dei mesi. Per tale motivo, la successiva fase di calibratura del modello S4b è stata eseguita in un ambiente di calcolo parallelo, riducendo i tempi d'esecuzione a meno di due settimane. L'adozione del Calcolo Parallelo e il conseguente abbattimento dei tempi d'esecuzione ha consentito, inoltre, di eseguire un'analisi sulla dinamica dell'Algoritmo Genetico in relazione a differenti funzioni di fitness.

Di seguito sono presentati i due nuovi modelli, gli esperimenti d'ottimizzazione e i risultati ottenuti sul caso di studio di Curti. Relativamente al modello S4b sono presentati, inoltre, i risultati su un caso di studio "idealizzato", definito per valutare gli effetti di differenti funzioni di fitness sulla dinamica dell'Algoritmo Genetico nella ricerca della soluzione ottimale.

5.2 Il modello SCIDDICA S4a

Definizione 5.2.1 (Definizione formale del modello SCIDDICA S4a). Il modello SCIDDICA S4a è formalmente definito nel seguente modo:

$$SCIDDICA\ S4a =< L, E, X, Q, P, \sigma, \gamma >$$

dove L, E, X, P e γ sono definiti esattamente come nella versione S3-hex dello stesso modello, presentato nel capitolo precedente. L'insieme degli stati dell'ae, Q, e la funzione di transizione, σ, sono invece definite nel seguente modo:

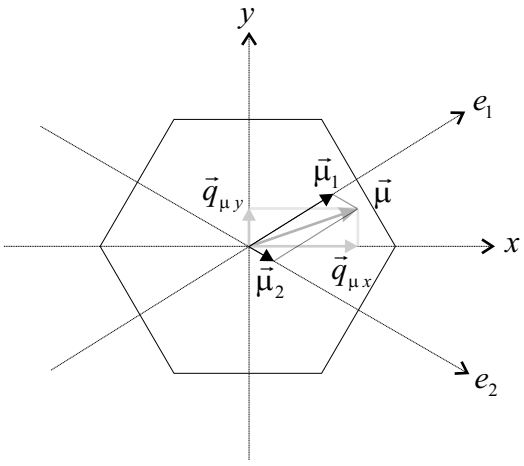


Figura 5.1: Definizione e composizione vettoriale degli indicatori inerziali $\overrightarrow{q_{\mu x}}$ e $\overrightarrow{q_{\mu x}}$: x e y sono gli assi del sistema di coordinate cartesiane ortogonali con origine nel centro della cella; $\overrightarrow{\mu}$ rappresenta la quantità di moto del detrito presente nella cella, espresso come somma vettoriale delle componenti $\overrightarrow{q_{\mu x}}$ e $\overrightarrow{q_{\mu x}}$ lungo gli assi x e y del sistema di coordinate cartesiane ortogonali e come somma delle componenti $\overrightarrow{\mu_1}$ e $\overrightarrow{\mu_2}$ lungo le due direzioni privilegiate, e_1 ed e_2, del reticolo esagonale.

$Q = Q_a \times Q_{th} \times Q_e \times Q_d \times Q_o^6 \times Q_{\mu x} \times Q_{\mu y}$ è l'insieme finito degli stati dell'ae, espresso come prodotto cartesiano dei sottostati considerati. Rispetto alla versione precedente, il modello S4a introduce i due sottostati $Q_{\mu x}$ e $Q_{\mu y}$ che rappresentano gli "indicatori" della quantità di moto del detrito, rispettivamente lungo le direzioni x e y di un sistema di riferimento cartesiano ortogonale con origine nel centro della cella (figura 5.1).

$\sigma : Q^7 \rightarrow Q$ è la funzione di transizione deterministica dell'ae, costituita dei seguenti processi elementari, applicati nello stesso ordine in cui sono presentati:

1. trasformazione interna T_1: attivazione ed effetto della mobilizzazione;
2. interazione locale I_1: flussi uscenti;
3. interazione locale I_2: aggiornamento dello spessore di detrito, dell'energia e della quantità di moto;
4. trasformazione interna T_2: perdita d'energia.

Le differenze rispetto alla versione precedente sono relative alle due interazioni locali I_1 e I_2, descritte più avanti nei dettagli.

All'inizio di ogni simulazione gli stati delle celle devono specificare le condizioni iniziali del sistema. I valori iniziali sono assegnati nel seguente modo:

- q_a è posto uguale alla quota (quota topografica dello strato roccioso più spessore dello strato erodibile – si veda la figura 4.8);
- q_{th} è zero ovunque tranne che nelle aree di innesco della frana, dove vale q_d;
- q_e è zero ovunque tranne che nelle aree di innesco della frana dove è uguale all'energia potenziale del detrito;
- q_d è lo spessore di suolo erodibile (nelle celle d'innesco $q_d = 0$);
- q_o, $q_{\mu x}$ e $q_{\mu y}$ sono zero ovunque.

La funzione di transizione è applicata alle celle dell'AC in modo che la configurazione del sistema cambi, e l'evoluzione della simulazione sia ottenuta.

5.2.1 Interazione locale I_1: determinazione dei flussi uscenti di detrito

L'interazione locale I_1, definita dalla funzione

$$\sigma_{I_1} : Q_a^7 \times Q_{th}^7 \times Q_e \times Q_{\mu x} \times Q_{\mu y} \rightarrow Q_o^6$$

determina i flussi di detrito uscenti dalla cella centrale verso le celle adiacenti. Essa si basa su un opportuno algoritmo di minimizzazione delle differenze, derivato dall'originario algoritmo proposto da Di Gregorio e Serra [59].

Al fine di enfatizzare gli effetti inerziali, nella versione S4a alcune direzioni sono privilegiate in accordo alla quantità di moto del detrito nella cella. Il vettore $\overrightarrow{\mu}$, definito come somma dei vettori $\overrightarrow{q_{\mu x}} \in Q_{\mu x}$ e $\overrightarrow{q_{\mu y}} \in Q_{\mu y}$, rappresenta la quantità di moto del flusso franante. Esso può essere espresso come la somma delle componenti $\overrightarrow{\mu_1}$ e $\overrightarrow{\mu_2}$, rispettivamente lungo le direzioni e_1 ed e_2 (che definiscono un sistema di coordinate non ortogonali con origine nel centro della cella), le quali sono al più perpendicolari a due lati adiacenti della cella (figura 5.1). I versi definiti da $\overrightarrow{\mu_1}$ e $\overrightarrow{\mu_2}$ rappresentano quelli "privilegiati" nel contesto dello spazio cellulare esagonale: lungo essi l'effetto del run-up deve essere maggiore, in modo da tenere in considerazione le proprietà inerziali del flusso detritico. Per definizione, $\mu_1 \geq \mu_2$: se $\mu_1 > \mu_2$, il verso definito da $\overrightarrow{\mu_1}$ è quello maggiormente privilegiato.

I moduli delle velocità v_{μ_1} e v_{μ_2}, rispettivamente lungo le direzioni e_1 ed e_2, possono essere dedotti dalle seguenti formule:

$$\begin{aligned} v_{\mu_1} &= \frac{\mu_1}{\rho A q_{th}(0)} \\ v_{\mu_2} &= \frac{\mu_2}{\rho A q_{th}(0)} \end{aligned}$$

essendo μ_1 e μ_2 i moduli delle componenti $\overrightarrow{\mu_1}$ e $\overrightarrow{\mu_2}$ della quantità di moto $\overrightarrow{\mu}$ e $\rho A q_{th}(0)$ la massa del detrito nella cella centrale. Dalle formula precedenti, per definizione di carico cinetico [120, 158], h_{k_1} e h_{k_2} sono calcolati come segue:

$$h_{k_1} = \frac{v_{\mu_1}^2}{2g} \tag{5.1}$$

$$h_{k_2} = \frac{v_{\mu_2}^2}{2g} \tag{5.2}$$

Si noti che $h_{k_1} \geq h_{k_2}$, essendo per definizione $\mu_1 \geq \mu_2$.

Al fine di privilegiare i versi definiti da $\overrightarrow{\mu_1}$ e $\overrightarrow{\mu_2}$, nel modello S4a le altezze (quota più spessore di detrito) delle celle adiacenti sono incrementate di un valore non negativo $w(i)$ $(i = 1, 2, \ldots, 6)$, calcolati nel seguente modo:

- $w(i) = h_{k_1} - h_{k_1} = 0$, se i è l'indice della cella adiacente lungo $\overrightarrow{\mu_1}$;
- $w(i) = h_{k_1} - h_{k_2} \geq 0$, se i è l'indice della cella adiacente lungo $\overrightarrow{\mu_2}$;
- $w(i) = h_{k_1}$, per i rimanenti versi, non privilegiati.

Quindi, l'altezza della cella maggiormente privilegiata (individuata da verso di $\overrightarrow{\mu_1}$) rimane inalterata; l'altezza dell'altra cella privilegiata (individuata da verso di $\overrightarrow{\mu_2}$) è incrementata di una quantità $w(i) \in [0, h_{k_1}]$; l'altezza delle rimanenti celle (lungo i versi non privilegiati) sono incrementate del valore massimo h_{k_1}. In tal modo, l'algoritmo di minimizzazione privilegerà, nel calcolo dei flussi di detrito, quelle celle le cui altezze rimangono inalterate o incrementate in misura minore.

In accordo alle precedenti considerazioni, nel modello S4a è stata sviluppata una strategia empirica a doppia fase al fine di considerare gli effetti inerziali di flussi di detrito rapidi.

Nella prima fase sono calcolati i flussi inerziali, $q'_o(0, i)$ $(i = 1, 2, \ldots, 6)$, introducendo l'"indebolimento" dei versi non privilegiati. L'algoritmo di minimizzazione è applicato alle seguenti quantità:

- $u(0) = q_a(0) + p_{adh}$;
- $m = r - p_{adh}$;
- $u(i) = q_a(i) + q_{th}(i) + w(i)$ $(i = 1, 2, \ldots, 6)$;

Precedentemente all'esecuzione dell'algoritmo di minimizzazione è eseguito un test preliminare: è calcolato l'angolo β_i tra la cella centrale, in termini d'altezza $u(0) + m$, e l'i-esima vicina, anch'essa in termini d'altezza $u(i)$, (si veda la figura 4.11) e le celle per le quali risulti $\beta_i < p_f$ sono eliminate dal calcolo dalla media e dalla fase di distribuzione.

La prima fase dell'algoritmo di minimizzazione è eseguita solo nel caso in cui risulti $q_{th}(0) - p_{adh} > 0$, essendo quest'ultima quantità quella che verrà effettivamente distribuita alle celle adiacenti. L'applicazione dell'algoritmo determina i flussi "virtuali" (o flussi di run-up) $f'(i)$ $(i = 1, 2, \ldots, 6)$, essendo in questa fase considerata come quantità distribuibile il run-up anzichè il detrito; gli effettivi flussi di detrito, $q'_o(0, i)$ $(i = 1, 2, \ldots, 6)$, sono ottenuti moltiplicando gli $f'(i)$ per il fattore di normalizzazione $v'_{nf} = q_{th}(0)/r$ e per il fattore di rallentamento p_r:

$$q'_o(0, i) = v'_{nf} f'(i) p_r$$

Il detrito residuo nella cella centrale, per effetto dei flussi uscenti $q'_o(0, i)$, è ridotto alla seguente quantità:

$$th(0) = q_{th}(0) - \sum_{i=1}^{6} q'_o(0, i)$$

Nella seconda fase di distribuzione, considerando l'eventuale detrito rimanente nella cella (cioè non distribuito durante la prima fase), sono calcolati i flussi "non inerziali" $q''_o(0, i)$ $(i = 1, 2, \ldots, 6)$. In questo caso l'algoritmo di minimizzazione è applicato alle seguenti quantità:

- $u(0) = q_a(0) + p_{adh}$;
- $m = r - p_{adh}$;
- $u(i) = q_a(i) + q_{th}(i)$ $(i = 1, 2, \ldots, 6)$;

Come nella prima fase, è eseguito lo stesso test preliminare e i flussi f'', calcolati dall'algoritmo di minimizzazione, sono normalizzati considerando il fattore $v''_{nf} = th(0)/r$ e il fattore di rallentamento p_r:

$$q''_o(0,i) = v''_{nf} f''(i) p_r$$

Al termine delle due fasi, i flussi totali $q_o(0,i)$ $(i = 1, 2, \ldots, 6)$ sono ottenuti come somma dei flussi inerziali e di quelli non inerziali:

$$q_o(0,i) = q'_o(0,i) + q''_o(0,i)$$

5.2.2 Interazione locale I_2: aggiornamento dello spessore di detrito, dell'energia e della quantità di moto

L'interazione locale I_2, definita dalla funzione

$$\sigma_{I_2} : (Q_{th} \times Q_e \times Q_o^6 \times Q_{\mu x} \times Q_{\mu y})^7 \rightarrow Q_{th} \times Q_e \times Q_{\mu x} \times Q_{\mu y}$$

aggiorna i valori di q_{th} e q_e dei sottostati spessore di detrito ed energia e delle componenti $\overrightarrow{q_{\mu x}}$ e $\overrightarrow{q_{\mu y}}$ della quantità di moto.

Il nuovo valore dello spessore di detrito nella cella, *new* q_{th}, è ottenuto considerando le variazioni dello spessore di detrito dovute ai flussi entranti e uscenti:

$$new\ q_{th} = q_{th}(0) + \sum_{i=1}^{6} (q_o(i,0) - q_o(0,i))$$

Come prima, il nuovo valore dell'energia, *new* q_e, è ottenuto considerando le variazioni dovute ai flussi entranti e uscenti:

$$new\ q_e = \left(q_{th}(0) - \sum_{i=1}^{6} q_o(0,i) \right) \cdot \left(k \frac{q_e(0)}{q_{th}(0)} \right) + \sum_{i=1}^{6} \left(q_o(i,0) \cdot \left(k \frac{q_e(i)}{q_{th}(i)} \right) \right)$$

dove $k = 2/\rho g A$ (si veda l'equazione 4.5).

Il valore delle componenti $\overrightarrow{q_{\mu x}}$ e $\overrightarrow{q_{\mu y}}$ della quantità di moto è ottenuto applicando il principio di conservazione dell'energia ai flussi entranti nella cella centrale. Sono calcolati i salti Δh_i $(i = 1, 2, \ldots, 6)$, che i flussi entranti compiono nel passaggio dall'i-esima vicina alla cella centrale, come differenza tra le altezze energentiche prima e dopo la distribuzione:

$$\Delta h_i = (q_a(i) + r(i)) - \left(q_a(0) + q_{th}(0) + \frac{q_o(i,0)}{v_{nf}} \right)$$

dove $r(i)$ è il run-up dell'i-esima cella vicina, mentre $q_o(i,0)/v_{nf}$ è il flusso di run-up associato al flusso di detrito $q_o(i,0)$ dall'i-esima vicina verso la cella centrale

($v_{nf} = q_{th}(0)/r(0)$). Sulla base dei dislivelli Δh_i è possibile calcolare il modulo delle velocità, v_i ($i = 1, 2, \ldots, 6$), dei flussi entranti nella cella cantrale:

$$v_i = \sqrt{2g\Delta h_i}$$

in cui g rappresenta l'accelerazione di gravità; in questa fase non sono tenuti in considerazione gli effetti dissipativi di cui si occupa la trasformazione interna T_2. Il modulo della quantità di moto associata all'i-esimo flusso entrante $q_o(i, 0)$ è quindi:

$$\mu_i = v_i q_o(i, 0)$$

Pertanto la quantità di moto della cella centrale è ottenuta come somma vettoriale delle quantità di moto associate ai flussi entranti e della quantità di moto del detrito nella cella centrale:

$$\overrightarrow{\mu} = \sum_{i=0}^{6} \overrightarrow{\mu_i}$$

essendo $\overrightarrow{\mu_0}$ la quantità di moto relativa al detrito nella cella centrale prima che siano considerati i contributi dei flussi entranti. Infine, le componenti $\overrightarrow{q_{\mu x}}$ e $\overrightarrow{q_{\mu y}}$ sono ottenute scomponendo la quantità di moto calcolata, $\overrightarrow{\mu}$, lungo gli assi del sistema di coordinate cartesiane ortogonali con origine nel centro della cella centrale (figura 5.1).

5.3 Ottimizzazione del modello S4a con Algoritmi Genetici

Come per i modelli precedenti della famiglia SCIDDICA, il confronto tra l'evento reale e l'evento simulato è stato eseguito considerando l'indicatore e_1, definito nel capitolo precedente, che fornisce una stima quantitativa sull'estensione areale della frana simulata rispetto alla reale. In questa fase è stato scelto come caso di studio l'evento di Curti (Sarno) del maggio 1998.

La fase di calibrazione preliminare, consistita, come per il modello S3-hex, nell'assegnare valori "ragionevoli" ai parametri del modello sulla base di precedenti simulazioni relative alla stessa area di studio, non ha consentito l'individuazione di un insieme di valori accettabili (nell'accezione definita nel capitolo precedente relativamente al caso di SCIDDICA S3-hex) per i parametri dell'AC. L'introduzione della quantità di moto nel modello ha, infatti, evidenziato il problema delle "direzioni privilegiate". In altri termini, se un flusso acquista una quantità di moto

troppo elevata l'effetto è quello di "incanalarsi" indiscriminatamente lungo una o entrambe le direzioni privilegiate (figura 5.1), risultando in un comportamento non realistico del sistema. Tale effetto poteva essere facilmente annullato aumentando il valore del parametro p_{rl}, responsabile della diminuzione del run-up a ogni passo di calcolo dell'AC. Questa operazione, tuttavia, presentava due problemi: 1) in ogni test eseguito si verificava l'annullamento degli effetti inerziali per cui il modello era stato sviluppato; 2) il confronto areale con l'evento reale risultava, comunque, non accettabile. Numerosi tentativi sono stati eseguiti variando anche altri parametri del modello (per esempio il fattore di rallentamento dei flussi p_r) ma sempre con risultati non soddisfacenti.

La complessità del nuovo modello, intesa come difficoltà nella ricerca di valori soddisfacenti per i suoi parametri, hanno suggerito l'impiego di una tecnica di calibrazione alternativa. Gli Algoritmi Genetici, presentati nel secondo capitolo, hanno dimostrato la loro validità nella ricerca di "buone soluzioni" per problemi anche molto complessi e per questo sono stati applicati alla calibratura dei parametri del modello S4a.

5.3.1 Implementazione di un Algoritmo Genetico per l'ottimizzazione di SCIDDICA S4a

I test preliminari discussi in precedenza, benchè non abbiano prodotto i risultati sperati, hanno dato importanti indicazioni sugli intervalli di variazione, $[a_i, b_i] \subset \mathbb{R}$, per i valori dei parametri, $p_i \in P$, dell'AC. La lista dei parametri ottimizzati tramite AG è riportata in tabella 5.1 insieme ai rispettivi intervalli di variazione; questi ultimi definiscono lo spazio di ricerca S_r dell'AG:

$$S_r = [a_1, b_1] \times [a_2, b_2] \ldots \times [a_5, b_5] = [0.001, 10] \times [0.1, 1] \times [0, 10] \times [0.001, 10] \times [0.001, 10]$$

Come descritto nel secondo capitolo, l'AG opera su una popolazione di n genotipi generata in modo casuale. Nell'originario modello di Holland, ogni individuo è rappresentato da una stringa di bit di lunghezza prefissata l, dipendente sia dal numero m di parametri $p_i \in P$ che devono essere ottimizzati, sia dal numero di bit k_i impiegati per la loro codifica. Risulta, infatti: $l = \sum_{i=1}^{m} k_i$. La codifica del primo parametro, $p_1^{(enc)}$, occupa i primi k_1 bit del genotipo; la codifica del secondo parametro, $p_2^{(enc)}$, occupa i bit tra $k_1 + 1$ e $k_1 + k_2$, e così via.

Gli individui sono valutati sulla base dell'errore, rispetto al caso reale, della simulazione eseguita adottando i valori dei parametri codificati nel genotipo. A

Parametro	Numero di bit	Intervallo di variazione	Miglior valore
$p_1 = p_{rl}$	$k_1 = 16$	$[a_1, b_1] = [0.001, 10]$	1.454124
$p_2 = p_r$	$k_2 = 16$	$[a_2, b_2] = [0.1, 1]$	0.759492
$p_3 = p_f$	$k_3 = 16$	$[a_3, b_3] = [0, 10]$	1.335622
$p_4 = p_{mt}$	$k_4 = 16$	$[a_4, b_4] = [0.001, 10]$	1.519273
$p_5 = p_{pef}$	$k_5 = 16$	$[a_5, b_5] = [0.001, 10]$	6.242842

Tabella 5.1: Lista dei parametri del modello SCIDDICA S4a ottimizzati con Algoritmi Genetici. La tabella riporta il numero di bit utilizzati per la codifica, l'intervallo di variazione e i valori relativi al miglior individuo evoluto.

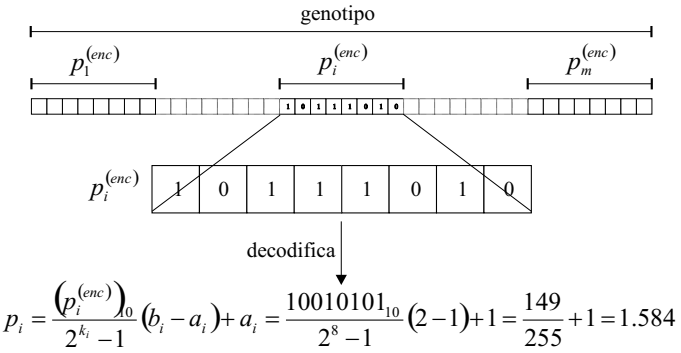


Figura 5.2: Esempio di decodifica di un parametro dal genotipo binario dell'AG utilizzato per l'ottimizzazione del modello SCIDDICA S4a: $p_i^{(enc)}$ è la codifica binaria dell'i-esimo parametro p_i del genotipo, k_i è il numero di bit utilizzati per la sua codifica, a_i e b_i sono gli estremi dell'intervallo in cui il parametro p_i può variare.

questo scopo, la funzione di fitness decodifica ogni individuo, fornendo il valore di ogni parametro p_i, tramite la seguente trasformazione lineare:

$$p_i = \frac{\left(p_i^{(enc)}\right)_{10}}{2^{k_i} - 1}(b_i - a_i) + a_i$$

dove $p_i^{(enc)}$ rappresenta la codifica binaria, $[a_i, b_i]$ l'intervallo di variazione e k_i il numero di bit utilizzati per la codifica del parametro p_i. La figura 5.2 mostra un esempio di decodifica di un parametro a partire dal genotipo.

Decodificato il genotipo e ottenuto il valore per ogni parametro dell'AC, è eseguita una simulazione. Il risultato è confrontato con la mappa dell'evento reale e

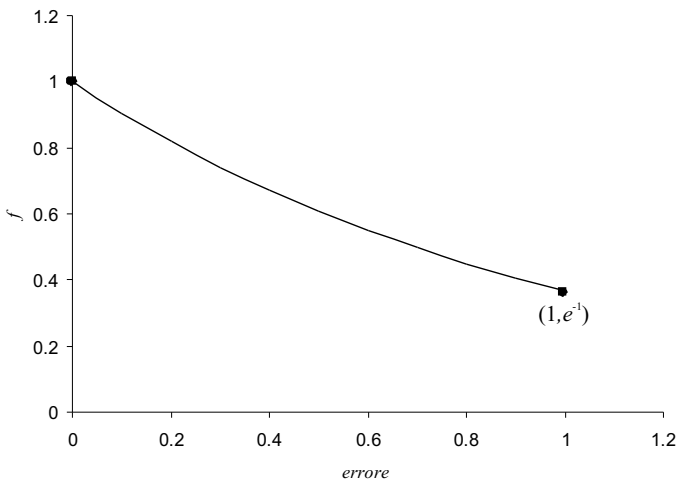


Figura 5.3: Funzione di fitness utilizzata per l'ottimizzazione del modello SCIDDICA S4a con Algoritmi Genetici.

valutato attraverso la seguente funzione:

$$1 - errore = \sqrt{\frac{\mu(R \cap S)}{\mu(R \cup S)}}$$

essendo R l'area affetta dalla frana reale, S l'area affetta dalla frana simulata, $\mu(R \cap S)$ e $\mu(R \cup S)$ rispettivamente la misura (in m^2) della loro intersezione e unione.

In generale, la funzione

$$errore = 1 - \sqrt{\frac{\mu(R \cap S)}{\mu(R \cup S)}}$$

vale 0 se la frana reale e la frana simulata sono perfettamente sovrapponibili, mentre vale 1 se sono completamente disgiunte. Per determinare il valore di fitness, f_i, per ogni genotipo, g_i, è stata scelta la funzione $f(g_i) = e^{-errore}$ (figura 5.3): f assume valori nell'intervallo $[e^{-1}, e^{0}]$. Ovviamente, i migliori individui sono quelli che mostrano valori di fitness più alti.

Valutato il valore di fitness di ogni individuo della popolazione, sono applicati gli operatori di selezione, crossover e mutazione, come nel classico modello di Holland descritto nel secondo capitolo. I valori delle probabilità di crossover e selezione

sono stati fissati a valori prestabiliti, mentre la probabilità di selezione è stata determinata in base al metodo della selezione proporzionale: $p_{selection,i} = f_i / \sum_{j=1}^{n} f_j$, essendo f_i il valore di fitness dell'i-esimo individuo ed n il numero di individui della popolazione.

L'applicazione dell'operatore di selezione e degli operatori genetici (crossover e mutazione) è stato iterato per un numero prefissato di generazioni, dando luogo a individui sempre più adatti (cioè abili nel fornire simulazioni sempre più simili all'evento reale).

5.3.2 Un esempio d'ottimizzazione sul caso di studio di Curti

La calibratura con Algoritmi Genetici del modello S4a è stata eseguita considerando come evento reale la frana di Curti, discussa nel capitolo precedente.

Durante tutti gli esperimenti, il parametro p_a è stato fissato a 1.25 m sulla base del dettaglio della mappa digitale delle quote (DEM – Digital Elevation Model), mentre il parametro p_{adh} è stato imposto pari a 0.001 m sulla base dei rilevamenti geologici di campo lungo il percorso della frana. Il parametro p_t, invece, non è stato impostato a un valore predefinito, nè ottimizzato tramite AG, dato che il suo valore può essere determinato solo *a posteriori* dall'analisi del comportamento complessivo del sistema.

I rimanenti $m = 5$ parametri sono stati, invece, codificati nel genotipo dell'AC utilizzando $k_i = 16$ bit per ognuno ($i = 1, 2, \ldots, 5$), e lo spazio di ricerca, S_r, dell'AG è stato definito sulla base delle precedenti simulazioni eseguite senza AG. La tabella 5.1 illustra i parametri soggetti a ottimizzazione insieme ai rispettivi intervalli di variazione.

La tabella 5.2 riporta i valori dei parametri dell'AG utilizzati nei test d'ottimizzazione; l'evoluzione temporale della fitness del miglior individuo e della fitness media dell'intera popolazione è illustrata in figura 5.4 come media sui risultati relativi ai cinque esperimenti eseguiti; la figura 5.5 mostra la simulazione dell'evento di Curti eseguita adottando i valori relativi al miglior individuo evoluto dall'AG.

I risultati sono stati molto soddisfacenti sia dal punto di vista qualitativo che da quello quantitativo. In particolare, nella zona di suddivisione del flusso alla base del pendio la capacità del detrito di avanzare in contropendenza (a partire da circa quota 200 m $s.l.m.$ – figura 5.5) è stata meglio simulata rispetto alle versioni precedenti. Infine, il valore dell'indicatore e_1 relativo alla simulazione in figura 5.5 è 0.8, che è il più alto mai ottenuto sul caso di studio di Curti. Tuttavia,

Parametro	Test 1	Test 2	Test 3	Test 4	Test 5
Probabilità di mutazione	0.00625	0.00625	0.00625	0.00625	0.00625
Probabilità di crossover	0.8	0.8	0.8	0.8	0.8
Dimensione della popolazione	30	30	30	30	30
Numero di generazioni	51	37	56	50	41
Seed	1	2	3	4	5
Miglior valore di fitness	0.81745	0.81504	0.81893	0.79772	0.80154

Tabella 5.2: Lista dei valori dell'Algoritmo Genetico utilizzati negli esperimenti d'ottimizzazione del modello SCIDDICA S4a. I migliori valori di fitness, ottenuti nei cinque esperimenti eseguiti, sono riportati nell'ultima riga.

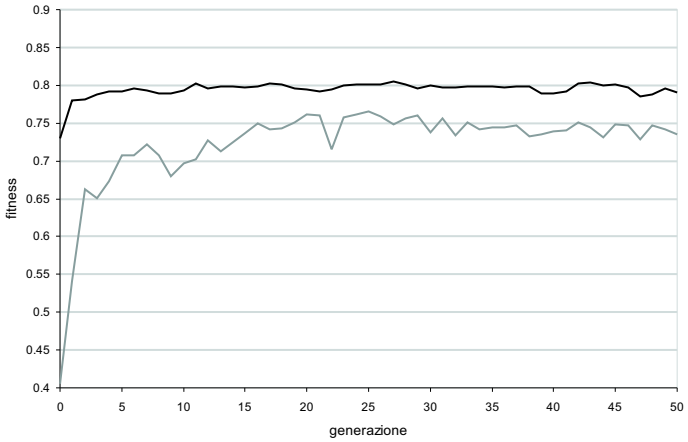


Figura 5.4: Evoluzione temporale della fitness come media sui cinque esperimenti eseguiti per l'ottimizzazione del modello SCIDDICA S4a: la linea nera rappresenta la fitness del miglior individuo, la linea grigia la fitness media dell'intera popolazione.

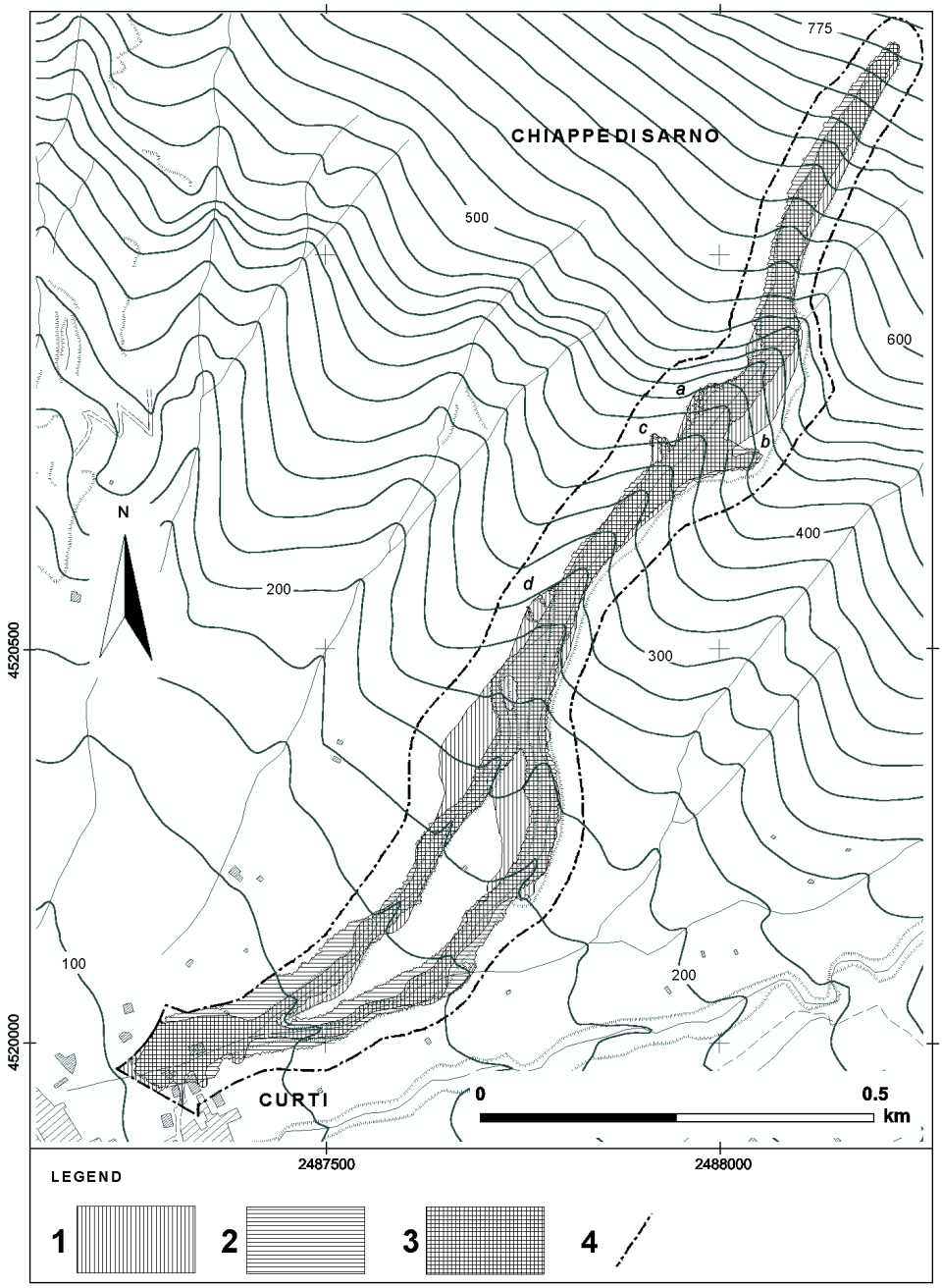


Figura 5.5: Confronto tra frana reale e frana simulata con il modello SCIDDICA S4a: (1) identifica le zone interessate dalla frana reale; (2) identifica le zone interessate dalla frana simulata; (3) identifica le zone interessate da entrambe le frane; (4) identifica i limiti dell'area relativa all'analisi eseguita in ambiente GIS; (a-d) identificano i punti d'innesco secondari. I valori dei parametri dell'AC utilizzati nella simulazione, evoluti tramite Algoritmi Genetici, sono riportati in tabella 5.1

l'esecuzione degli esperimenti d'ottimizzazione ha richiesto più di cinque mesi di calcolo su un Personal Computer di classe Pentium IV a 1400 MHz; pertanto, per il modello S4b, la stessa fase di calibratura è stata eseguita in ambiente di calcolo parallelo.

5.4 Il modello SCIDDICA S4b

Definizione 5.4.1 (Definizione formale del modello SCIDDICA S4b). Il modello SCIDDICA S4b è formalmente definito nel seguente modo:

$$SCIDDICA\ S4b =< L, E, X, Q, P, \sigma, \gamma >$$

dove L, E, X e Q sono definiti esattamente come nella versione S4a, presentato nella sezione 5.2. L'insieme dei parametri dell'ae, P, la funzione di transizione, σ, e la funzione d'innesco delle nicchie, γ, sono invece definite nel seguente modo:

$P = \{p_a, p_t, p_{adh}, p_f, p_r, p_{rl}, p_{mt}, p_{er}, p_{ltt}, p_{if}\}$ è l'insieme dei parametri dell'AC. Rispetto alla versione S4a, il modello S4b introduce due nuovi parametri: p_{ltt} e p_{if}: il parametro p_{ltt} rappresenta la soglia, in termini di spessore di detrito, per l'attivazione delle sorgenti secondarie; il parametro p_{if} rappresenta il fattore inerziale (gli effetti inerziali vengono amplificati, nel caso $p_{if} > 1$, o inibiti, nel caso $p_{if} < 1$).

$\sigma : Q^7 \rightarrow Q$ è la funzione di transizione deterministica dell'ae, costituita dei seguenti processi elementari, applicati nello stesso ordine in cui sono presentati:

1. trasformazione interna T_1: attivazione ed effetto della mobilizzazione;
2. interazione locale I_1: flussi uscenti;
3. interazione locale I_2: aggiornamento dello spessore di detrito, dell'energia e della quantità di moto;
4. trasformazione interna T_2: perdita d'energia.

$\gamma : E \times \mathbb{N} \times Q_d \times Q_{th} \times Q_e \rightarrow Q_a \times Q_d \times Q_{th} \times Q_e$ è la funzione d'attivazione delle sorgenti d'innesco della frana. Le sorgenti principali s'innescano al primo passo; ulteriori sorgenti possono innescarsi successivamente a passi prefissati dell'AC o per effetti destabilizzanti causati dal passaggio della massa franante. $\mathbb{N}$ è l'insieme dei numeri naturali.

Le uniche differenze rispetto alla versione S4a sono relative all'interazione locale I_1 e alla funzione d'attivazione delle sorgenti d'innesco γ, descritte più avanti nei dettagli.

5.4.1 Interazione locale I_1: determinazione dei flussi uscenti di detrito

L'interazione locale I_1, definita dalla funzione

$$\sigma_{I_1} : Q_a^7 \times Q_{th}^7 \times Q_e \times Q_{\mu x} \times Q_{\mu y} \rightarrow Q_o^6$$

determina i flussi di detrito uscenti dalla cella centrale verso le celle adiacenti. Come nei casi precedenti, essa si basa su un opportuno algoritmo di minimizzazione delle differenze, derivato dall'originario algoritmo proposto da Di Gregorio e Serra [59].

Al fine di enfatizzare gli effetti inerziali, nella versione S4a alcune direzioni sono privilegiate in accordo alla quantità di moto del detrito nella cella.

Come nel modello S4a, sono calcolate le quantità h_{k_1} (equazione 5.1) e h_{k_2} (equazione 5.2). Successivamente, allo scopo di privilegiare i versi definiti da $\overrightarrow{\mu_1}$ e $\overrightarrow{\mu_2}$ (figura 5.1), in S4b le altezze (in termini di quota più spessore di detrito) delle celle adiacenti alla cella centrale sono incrementate delle quantità $w(i)$ ($i = 1, 2, \ldots, 6$), calcolate come segue:

- $w(i) = -p_{if} h_{k_1}$, essendo i l'indice della cella adiacente lungo $\overrightarrow{\mu_1}$;
- $w(7-i) = p_{if} h_{k_1}$, essendo $7-i$ l'indice della cella adiacente opposta a $\overrightarrow{\mu_1}$;
- $w(i) = -p_{if} h_{k_2}$, essendo i l'indice della cella adiacente lungo $\overrightarrow{\mu_2}$;
- $w(7-i) = p_{if} h_{k_2}$, essendo $7-i$ l'indice della cella adiacente opposta a $\overrightarrow{\mu_2}$;
- $w(i) = w(7-i) = 0$, essendo i e $7-i$ gli indici delle celle lungo la direzione rimanente.

Quindi, l'altezza della cella adiacente alla cella centrale maggiormente privilegiata (individuata dal verso di $\overrightarrow{\mu_1}$) è diminuita di un valore massimo dato da $w(i) = -p_{if} h_{k_1}$. L'altezza della cella adiacente alla cella centrale e opposta a quella maggiormente privilegiata (lungo la direzione e_1) è incrementata del valore massimo dato da $w(7-i) = p_{if} h_{k_1}$. Analogamente, le altezze delle celle lungo la direzione e_2 vengono alterate sommano la quantità $w(i) = -p_{if} h_{k_2}$ (per la cella individuata dal verso del vettore $\overrightarrow{\mu_2}$) e $w(7-i) = p_{if} h_{k_2}$ (per quella opposta).

Le altezza delle ultime due celle adiacenti rimangono inalterate. In questo modo l'algoritmo di minimizzazione privilegerà, nella distribuzione del detrito, le celle le cui altezze sono ridotte.

In accordo con le precedenti considerazioni, i flussi inerziali sono calcolati applicando l'algoritmo di minimizzazione alle seguenti quantità:

- $u(0) = q_a(0) + p_{adh}$;
- $m = r - p_{adh}$;
- $u(i) = q_a(i) + q_{th}(i) + w(i)$ $(i = 1, 2, \ldots, 6)$;

Prima dell'applicazione dell'algoritmo di minimizzazione è eseguito il test preliminare come descritto per i modelli S3-hex ed S4a. Inoltre, l'algoritmo di minimizzazione è applicato solo nel caso in cui risulti $q_{th}(0) - p_{adh} > 0$, essendo quest'ultima quantità quella che verrà effettivamente distribuita alle celle adiacenti.

L'algoritmo determina i flussi "virtuali" $f(i)$ $(i = 1, 2, \ldots, 6)$, essendo in questa fase considerata come quantità distribuibile il run-up anzichè il detrito; gli effettivi flussi di detrito, $q_o(0, i)$ $(i = 1, 2, \ldots, 6)$, sono ottenuti moltiplicando gli $f(i)$ per il fattore di normalizzazione $v_{nf} = h/r$ e per il fattore di rallentamento p_r:

$$q_o(0, i) = v_{nf} f(i) p_r$$

5.4.2 La funzione d'attivazione delle sorgenti d'innesco γ

La funzione d'attivazione delle sorgenti, responsabile dell'innesco dell'evento, è definita nel seguente modo:

$$\gamma : E \times \mathbb{N} \times Q_d \times Q_{th} \times Q_e \rightarrow Q_a \times Q_d \times Q_{th} \times Q_e$$

essendo $E \subset L$ l'insieme delle celle che specificano i punti d'innesco ed $\mathbb{N}$ l'insieme dei numeri naturali, corrispondenti ai passi dell'AC.

Le sorgenti principali s'innescano al primo passo dell'AC, mentre ulteriori sorgenti, dette secondarie, possono innescarsi successivamente, a passi prefissati dell'AC o per effetti destabilizzanti causati dal passaggio della massa franante. In quest'ultimo caso, un valore di soglia, specificato dal parametro p_{ltt}, determina lo spessore di detrito necessario all'attivazione dei siti d'innesco secondari. Il controllo sullo spessore di detrito franante è eseguito in un insieme prefissato di celle che appartengono al "transetto", localizzato trasversalmente lungo il percorso della frana e riferito a un dato sito d'innesco. L'attivazione del sito d'innesco è determinato,

dunque, dal valore dello spessore di detrito presente nel transetto. In altri termini, la condizione d'attivazione per l'i-esimo sito è $q_{th} \geq p_{ltt}$, dove q_{th} è valutato nelle celle dell'i-esimo transetto.

Quando una sorgente s'innesca, come per i modelli precedenti, sono considerate le seguenti variazioni:

$$\begin{aligned} new\ q_a &= q_a(0) - q_d(0) \\ new\ q_d &= q_d(0) - q_d(0) = 0 \\ new\ q_{th} &= q_{th}(0) + q_d(0) \\ new\ r &= r + q_d(0) \end{aligned}$$

e l'energia è ricalcolata in riferimento allo spessore di detrito e al run-up in accordo alla seguente formula:

$$new\ q_e = \frac{1}{k}(q_{th}(0) + q_d(0))(r + q_d(0)) = \frac{1}{k}(q_{th}(0) + q_d(0))\left(k\frac{q_e(0)}{q_{th}(0)} + q_d(0)\right)$$

essendo $k = 2/\rho g A$ (equazione 4.5).

5.5 Ottimizzazione del modello S4b con Algoritmi Genetici Paralleli

Come nel caso della versione S4a, l'ottimizzazione della modello S4b è stata eseguita sul caso di studio di Curti.

Durante tutti gli esperimenti, il parametro p_a è stato fissato a 1.25 m sulla base del dettaglio della mappa digitale delle quote (DEM – Digital Elevation Model); il parametro p_t, invece, non è stato impostato a un valore predefinito, nè ottimizzato tramite AG, dato che il suo valore può essere determinato solo *a posteriori* dall'analisi del comportamento complessivo del sistema. I rimanenti $m = 8$ parametri sono stati codificati nel genotipo dell'AG direttamente come valori reali, considerando popolazioni di 200 individui. La lista dei parametri ottimizzati tramite AG è riportata in tabella 5.3.

I test di calibratura preliminari, come discussi per il caso del modello S4a, benchè non abbiano prodotto i risultati sperati, hanno dato importanti indicazioni sugli intervalli di variazione, $[a_i, b_i] \subset \mathbb{R}$, per i valori dei parametri, $p_i \in P$, dell'AC:

Parametri	Intervallo di variazione	Miglior valore
$p_1 = p_{rl}$	$[a_1, b_1] = [0.001, 15]$	2.5021
$p_2 = p_{adh}$	$[a_2, b_2] = [0.0001, 0.01]$	0.007035402
$p_3 = p_r$	$[a_3, b_3] = [0.001, 1]$	0.1942762
$p_4 = p_f$	$[a_4, b_4] = [0.001, 15]$	0.1426551
$p_5 = p_{mt}$	$[a_5, b_5] = [0.001, 15]$	0.0494169
$p_6 = p_{pef}$	$[a_6, b_6] = [0.001, 5]$	1.0253
$p_7 = p_{ltt}$	$[a_7, b_7] = [0.1, 5]$	0.3715202
$p_8 = p_{if}$	$[a_8, b_8] = [1, 5]$	2.321504

Tabella 5.3: Lista dei parametri del modello SCIDDICA S4b ottimizzati con Algoritmi Genetici. La tabella riporta l'intervallo di variazione e i valori relativi al miglior individuo evoluto.

questi intervalli definiscono lo spazio di ricerca S_r dell'AG:

$$S_r = [a_1, b_1] \times [a_2, b_2] \ldots \times [a_8, b_8] = [0.001, 15] \times [0.0001, 0.01] \times \\ \times [0.001, 1] \times [0.001, 15] \times [0.001, 15] \times [0.001, 5] \times [0.1, 5] \times [1, 5] \quad (5.3)$$

L'AG utilizzato per l'ottimizzazione del modello S4b è di tipo steady-state elitistico, poichè a ogni generazione solo i peggiori individui della popolazione (nel nostro caso i peggiori 15) sono rimpiazzati con i discendenti ottenuti tramite crossover e mutazione. I rimanenti (185) individui, necessari a formare la nuova popolazione, sono copiati dalla vecchia scegliendo i migliori. L'operatore di selezione è il "binary tournament without replacement", che consiste in una serie di "tornei" in cui due individui della vecchia popolazione sono selezionati in modo casuale e il vincitore è scelto in accordo a una prefissata probabilità che deve essere maggiore per l'individuo con fitness più alta. Nel nostro caso tale probabilità è stata fissata a 0.6. Inoltre, poichè è stata scelta la variante "without replacement", gli individui non possono essere selezionati più di una volta per la riproduzione. Infine, gli operatori genetici utilizzati sono i classici operatori di crossover e mutazione dell'originario modello di Holland. La lista completa delle specifiche dell'AG utilizzato per l'ottimizzazione di SCIDDICA S4b è riportata in tabella 5.4.

La scelta di un'appropriata funzione di fitness è essenziale per valutare la bontà di una simulazione. Nel lavoro di ottimizzazione sul modello S4b sono state considerate due differenti funzioni di fitness: f_1, che semplicemente confronta le estensioni areali della frana simulata e della frana reale, ed f_2, che considera sia l'estensione areale che le modificazioni dello strato erodibile. Tuttavia, informazioni di qualità

Caratteristiche e parametri	Specifiche
Codifica del genotipo	Stringa di numeri reali
Lunghezza del genotipo	8
Dimensione della popolazione	200
Schema di rimpiazzamento	Steady-state ed elitistico
Numero di individui rimpiazzati a ogni generazione	15
Operatore di selezione	Binary tournament without replacement
Probabilità di vittoria per l'individuo con fitness più alta	0.6
Operatore di crossover	A singolo punto
Probabilità di crossover	0.8
Operatore di mutazione	Variazione casuale del valore dei geni
Probabilità di mutazione	0.125

Tabella 5.4: Lista delle caratteristiche e dei parametri dell'Algoritmo Genetico utilizzato per l'ottimizzazione del modello SCIDDICA S4b.

sull'erosione dello strato detritico, così come sulla deposizione del detrito lungo il percorso della frana, sono solitamente difficili da reperire. Nel caso dell'evento di Curti, così come per gli altri casi sul versante di Pizzo d'Alvano, solo l'estensione areale era nota con adeguato dettaglio. Di conseguenza solo la funzione di fitness f_1 è stata impiegata nella fase di calibratura sul caso reale. Tuttavia un confronto tra le due funzioni di fitness è stato analizzato su un caso di studio idealizzato; i risultati sono presentati nel paragrafo 5.6

La funzione di fitness f_1 e definita nel seguente modo:

$$f_1 = \frac{\mu(R \cap S)}{\mu(R \cup S)} \tag{5.4}$$

dove R ed S identificano rispettivamente l'estensione areale della frana reale e della frana simulata, mentre $\mu(R \cap S)$ e $\mu(R \cup S)$ sono rispettivamente le misure (in m^2) della loro intersezione e della loro unione.

La funzione f_1 assume valori nell'intervallo $[0, 1]$: essa vale 0 se le due frane sono completamente disgiunte, essendo $\mu(R \cap S) = 0$; vale 1 se esse sono perfettamente sovrapponibili, essendo in questo caso $\mu(R \cap S) = \mu(R \cup S)$. Così, l'obbiettivo per l'AG è trovare un insieme di valori per i parametri dell'AC tali che massimizzino f_1.

5.5.1 Implementazione parallela e performance

Per valutare un qualsiasi individuo dell'AG è necessario eseguire un'intera simulazione (15000 passi di calcolo dell'AC) e confrontare poi il risultato con il caso di studio considerato. A seconda dell'architettura di calcolo utilizzata, tale operazione può richiedere da alcuni minuti a qualche ora. Per esempio, su una workstation Pentium 4 a 1400 MHz, tale operazione richiede all'incirca 15 minuti. Così, se in un esperimento si 200 generazioni, in ognuna devono essere valutati 15 individui, il tempo richiesto per 200 supera i 31 giorni. Inoltre và cosiderato che il tempo di calcolo può aumentare ulteriormente al crescere dell'estensione dell'area in cui si sviluppa il fenomeno e del dettaglio delle mappe digitali del suolo.

Gli AG sono, tuttavia, algoritmi "imbarazzantemente paralleli", nel senso che è semplice realizzare implementazioni che possano sfruttare efficientemente più CPU. Il Calcolo Parallelo può rappresentare, pertanto, un efficace strumento per ridurre significativamente i tempi d'esecuzione degli AG. Diversi esempi di AG Paralleli sono stati proposti in letteratura [175, 25, 2]. Tra questi, il più semplice è rappresentato dal modello "Master-Slave" in cui un processore (il master) esegue i passi dell'AG (selezione, rimpiazamento della popolazione, crossover e mutazione), mentre altri (gli slave) valutano la fitness degli individui.

Sono state realizzate numerose implementazioni parallele di AG, alcune delle quali sono liberamente disponibili su Internet. Tra esse, il PGAPack è stato scelto per l'ottimizzazione del modello S4b poichè è disponibile per diversi sistemi operativi come *Linux e *BSD, e fornisce un supporto completo per una varietà di schemi di selezione e rimpiazzamento della popolazione e operatori genetici. Inoltre, è possibile utilizzare il PGAPack sia su architetture a memoria distribuita che a memoria condivisa utilizzando l'interfaccia MPI (Message Passing Interface) [146, 75] per lo scambio di informazioni tra i processori.

Un cluster di classe Beowulf di workstation Linux è stato utilizzato come macchina parallela. I 16 nodi monoprocessore sono interconnessi tramite una rete Ethernet LAN (Local Area Network) a 100 Mbs. Ogni nodo è un comune Personal Computer con processore Pentium 4 a 1400 MHz, 512 MB di memoria RAM, sistema operativo Red Hat Linux 7.2 e compilatore GCC 2.96.

Dimezzare il tempo d'esecuzione sequenziale ogni volta che si raddoppia il numero di processori utilizzati rappresenta la *performance* ideale di un programma parallelo su una data macchina parallela. Lo speed-up è definito come rapporto tra tempo d'esecuzione sequenziale e tempo d'esecuzione parallelo. Quindi, se N_p rappresenta il numero di processori impiegati, il tempo parallelo ideale è

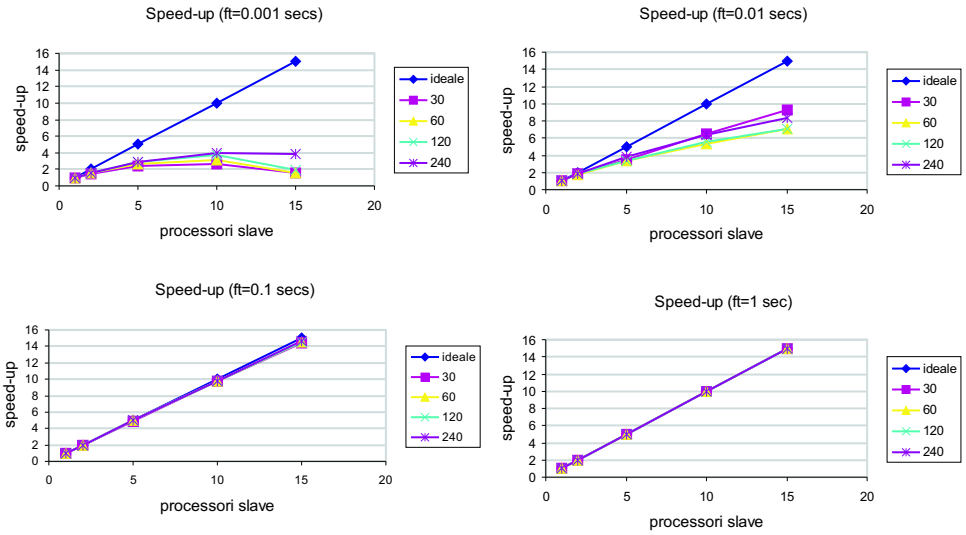


Figura 5.6: Speed-up del PGAPack sul cluster Beowulf utilizzato per gli esperimenti di ottimizzazione del modello SCIDDICA S4b. I grafici si riferiscono a esperimenti con differenti numeri d'individui e differenti tempi d'esecuzione della funzione di fitness. Lo speed-up lineare è ottenuto per $f_t > 0.1$ secondi.

$T_p = T_s/N_p$, dove T_s rappresenta il tempo d'esecuzione sequenziale. Come conseguenza, lo speed-up ideale è $T_s/T_p = N_p$: in tal caso si dice che il programma parallelo scala linearmente col numero di processori.

Le prestazioni del PGAPack sono state preliminarmente misurate in termini di speed-up, eseguendo 100 generazioni di 4 semplici AG (come originariamente proposti da Holland), rispettivamente caratterizzati da popolazioni di $n = 30$, 60, 120 e 240 individui. Per ogni AG, è stata considerata una funzione di fitness "fittizia" che semplicemente aspetta per intervalli di tempo $f_t = 0.001$, 0.01, 0.1 e 1 secondi. Per ogni combinazione di n ed f_t è stata eseguita un'applicazione dell'AG considerando $N_p = 1$, 2, 5 ,10 e 15 processori slave, e i tempi d'esecuzione sequenziale e paralleli sono stati stimati in modo da determinare lo speed-up. I risultati hanno dimostrato la scalabilità lineare del PGAPack sul cluster Beowulf considerato per $f_t > 0.1$ secondi (figura 5.6).

Dato che il tempo necessario per la valutazione di una soluzione candidata del modello S4b è generalmente maggiore (il tempo di calcolo per una simulazione dell'evento di Curti è dell'ordine delle decine di minuti), l'architettura parallela considerata garantisce uno speed-up lineare, risultando così la soluzione ottimale

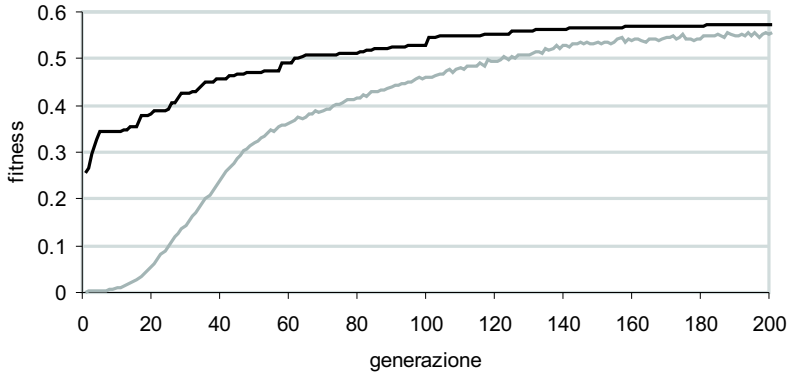


Figura 5.7: Evoluzione temporale della fitness come media sui cinque esperimenti eseguiti per l'ottimizzazione del modello SCIDDICA S4b: la linea nera rappresenta la fitness del miglior individuo, la linea grigia la fitness media dell'intera popolazione.

in termini di rapporto prezzo/prestazioni per l'applicazione di Algoritmi Genetici Paralleli.

5.5.2 Esperimenti e risultati sul caso di studio di Curti

Considerando l'evento di Curti, sono state eseguite cinque applicazioni dell'AG, ognuna di 200 generazioni e differente popolazione iniziale. In complesso, il processo d'ottimizzazione è durato meno di 11 giorni (si noti che lo stesso esperimento sarebbe durato più di cinque mesi su una macchina sequenziale). L'evoluzione temporale della fitness è graficamente riportata in figura 5.7, in termini di risultati medi sui cinque esperimenti eseguiti. I valori dei parametri dell'AC corrispondenti al miglior individuo sono elencati in tabella 5.3. La simulazione, ottenuta adottando i valori dei parametri relativi al miglior individuo, è riportata in figura 5.8.

I risultati sono abbastanza soddisfacenti sia dal punto di vista quantitativo che qualitativo. In particolare, nella zona della suddivisione del flusso alla base del pendio, il valore del detrito che riesce ad avanzare in contropendenza (a partire da circa quota 200 m $s.l.m.$ – figura 5.8) è meglio simulata rispetto alle precedenti versioni del modello che non tenevano in considerazione gli effetti inerziali del flusso detritico. Inoltre, il valore dell'indicatore e_1 relativo alla simulazione eseguita

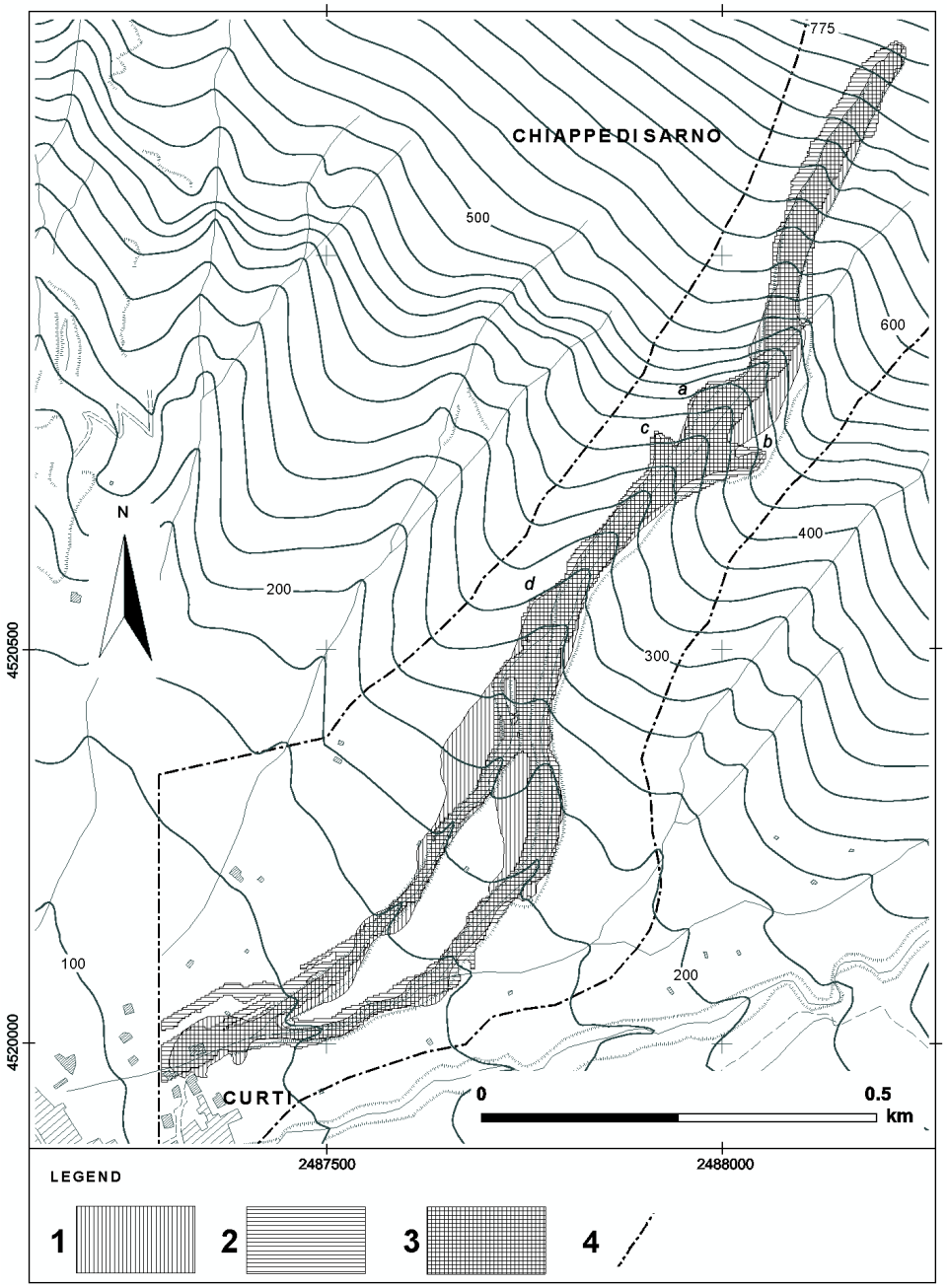


Figura 5.8: Confronto tra frana reale e frana simulata con il modello SCIDDICA S4b: (1) identifica le zone interessate dalla frana reale; (2) identifica le zone interessate dalla frana simulata; (3) identifica le zone interessate da entrambe le frane; (4) identifica i limiti dell'area relativa all'analisi eseguita in ambiente GIS; (a-d) identificano i punti d'innesco secondari. I valori dei parametri dell'AC utilizzati nella simulazione , evoluti tramite Algoritmi Genetici, sono riportati in tabella 5.4.

adottando i parametri del miglior individuo evoluto dall'AG è pari a 0.77, che è il valore più alto mai ottenuto per il modello S4b.

5.6 Considerazioni sulla dinamica dell'Algoritmo Genetico

Gli esperimenti di ottimizzazione eseguiti sul modello S4b hanno messo in evidenza la capacità dell'AG di evolvere individui caratterizzati da fitness relativamente alta (rispetto al miglior valore ottenuto nell'esperimento nel suo complesso) sin dalle prime generazioni (figura 5.7), nonostante la popolazione iniziale fosse stata generata in maniera casuale su uno spazio di ricerca piuttosto grande (equazione 5.3). Questo comportamento è dovuto, almeno in parte, alle presenza di forti vincoli morfologici lungo il percorso del flusso detritico. Infatti, l'analisi di numerose simulazioni eseguite adottando come parametri dell'AC i valori relativi ai migliori individui evoluti nelle prime generazioni (che definiscono insiemi alternativi dei valori dei parametri) hanno evidenziato comportamenti molto simili (in termini dei risultati delle simulazioni), in particolar modo nella parte superiore del canale dove il vincolo morfologico è più forte. Questo comportamento, che a una prima analisi potrebbe apparire un fatto decisamente positivo, potrebbe invece influenzare negativamente la ricerca di buone soluzioni e indirizzare l'AG verso zone dello spazio di ricerca caratterizzate da uno o più ottimi locali. In queste condizioni non è detto, infatti, che una buona simulazione corrisponda necessariamente a un buon insieme di valori per i parametri del modello, dato che il risultato potrebbe essere significativamente condizionato dalla presenza del vincolo morfologico. Il rischio è quello di ottenere un insieme di parametri che funzionino adeguatamente sul caso di studio considerato nella fase d'ottimizzazione, ma che fallisca nell'applicazioni a casi di studio equivalenti che evolvono in differenti contesti morfologici. Si capisce dunque che la convergenza verso un ottimo locale potrebbe creare grossi problemi qualora il modello fosse applicato alla definizione di mappe di rischio: esso potrebbe dar luogo a risultati del tutto inconsistenti. Una possibile soluzione al problema potrebbe consistere nell'eseguire l'ottimizzazione contemporaneamente su più casi di studio, ma la disponibilità dei dati non sempre lo consente e i tempi di calcolo potrebbero crescere sino a limiti non accettabili. Diventa, allora, indispensabile individuare una strategia d'ottimizzazione efficiente che riduca il più possibile il rischio di imbattersi in ottimi locali.

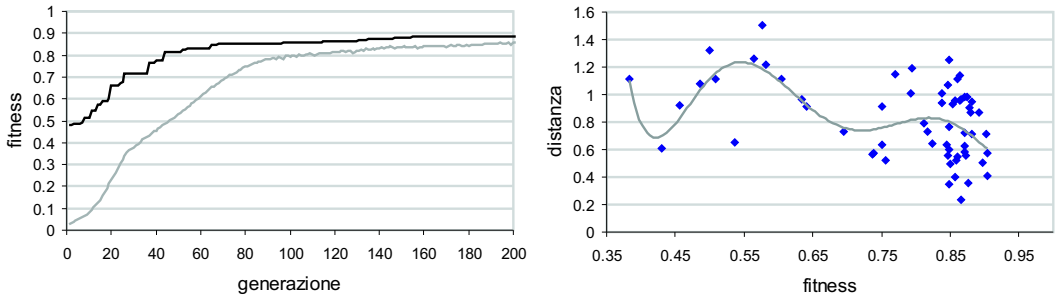
Per meglio indagare la dinamica dell'AG, è stato considerato un caso di stu-

dio idealizzato. E' stata prima eseguita una simulazione considerando un insieme prefissato di parametri, P_{opt}, selezionato tra i migliori ottenuti nella fase di calibrazione sul caso di studio di Curti. Il risultato della simulazione è stato assunto come evento reale, o meglio "pseudo reale", per gli esperimenti d'ottimizzazione successivi. In questo modo, l'ottimo globale è noto (esso è infatti P_{opt}) e la dinamica dell'AG può essere analizzata considerando le distanze euclidee tra gli individui evoluti e l'ottimo globale P_{opt} nello spazio di ricerca S_r. Così, in una rappresentazione grafica dell'andamento della distanza sulla fitness, se la distanza diminuisce significativamente con la fitness, allora la tendenza della dinamica dell'AG è verso l'ottimo globale; al contrario, se a) si osservano forti oscillazioni o b) le distanze aumentano all'aumentare della fitness, si può concludere che a) l'AG oscilla tra diversi ottimi locali oppure b) converge verso una soluzione non ottimale.

Sono stati, quindi, eseguiti quattro esperimenti d'ottimizzazione del modello SCIDDICA S4b con l'AG già applicato all'ottimizzazione dello stesso modello sul caso di studio di Curti. In ogni esperimento è stata generata in maniera casuale una popolazione iniziale ed f_1 (equazione 5.4) è stata scelta come funzione di fitness. I risultati sono illustrati in figura 5.9. Come è possibile notare in figura 5.9b, che riporta l'andamento della distanza tra gli individui evoluti dall'AG e P_{opt}, si osservano forti oscillazioni, in particolar modo quando sono raggiunti valori alti di fitness. Questo comportamento è riconducibile al punto a) discusso in precedenza; pertanto la ricerca della soluzione ottimale, P_{opt}, potrebbe risultare fortemente compromessa dalla presenza di uno o più ottimi locali nello spazio di ricerca S_r. Nondimeno, l'AG ha evidenziato una leggera tendenza verso l'ottimo globale (si noti la linea di tendenza in figura 5.9b) ed è stato capace, nonostante l'andamento oscillatorio, di evolvere soluzioni con alti valori di fitness: $f_1 = 0.9$ per il miglior individuo (figura 5.9a).

Nel tentativo di ridurre il numero di ottimi locali nello spazio di ricerca dell'AG, è stata definita una nuova funzione funzione di fitness, f_2, che valuta la bontà di una simulazione sulla base sia del confronto areale, sia sulla base del confronto sull'erosione dello strato detritico lungo il percorso del flusso. L'ipotesi è che, inglobando anche l'informazione relativa all'erosione, la nuova funzione di fitness possa meglio discriminare tra individui più o meno abili, riducendo, di conseguenza, il numero di ottimi locali nello spazio di ricerca. La funzione di fitness f_2 è definita nel seguente modo:

$$f_2 = \frac{\mu(R \cap S)}{\mu(R \cup S)} \cdot \left(1 - \frac{\sum_L |r(R) - r(S)|}{\sum_L |r(R) + r(S)|}\right) = f_1 \cdot \left(1 - \frac{\sum_L |r(R) - r(S)|}{\sum_L |r(R) + r(S)|}\right)$$



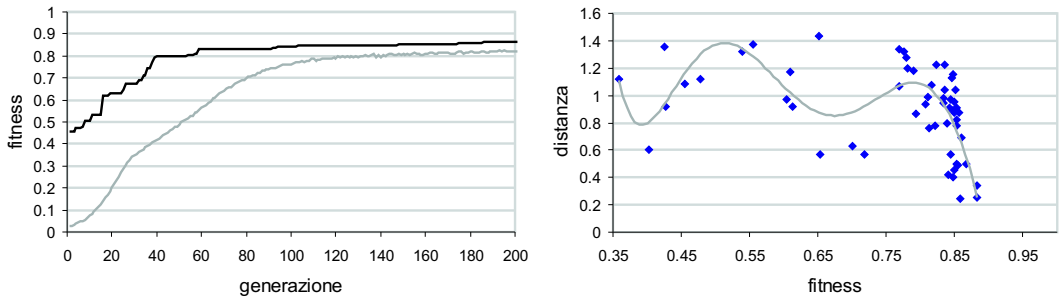
(a) (b)

Figura 5.9: Risultati degli esperimenti di ottimizzazione sul caso di studio idealizzato adottando f_1 come funzione di fitness: (a) mostra l'evoluzione temporale della fitness (in nero la fitness del miglior individuo, in grigio la fitness media dell'intera popolazione), ottenuta come media sui quattro esperimenti eseguiti; (b) mostra la distanza tra gli individui evoluti dall'Algoritmo Genetico e l'ottimo globale P_{opt} (in grigio la linea di tendenza).

dove R ed S identificano, rispettivamente, l'estensione areale della frana reale e della frana simulata, $\mu(R \cap S)$ e $\mu(R \cup S)$ le misure (in m^2) della loro intersezione e della loro unione, $r(R)$ ed $r(S)$ lo spessore dello strato detritico eroso in una generica cella dello spazio cellulare L per effetto della frana reale e simulata.

Come f_1, la funzione di fitness f_2 assume valori nell'intervallo $[0, 1]$. In particolare $f_2 = 1$ quando l'evento simulato e l'evento reale sono perfettamente sovrapponibili, essendo $\mu(R \cap S) = \mu(R \cup S)$, e l'errore relativo all'erosione dello strato detritico è nullo in ogni cella dello spazio cellulare, essendo $\sum_L |r(R) - r(S)| = 0$.

Gli esperimenti eseguiti considerando f_1 come funzione di fitness sono stati ripetuti adottando la nuova funzione f_2 sullo stesso caso di studio idealizzato (figura 5.10). Come è possibile notare in figura 5.10b, che rappresenta la distanza tra gli individui evoluti dall'AG e P_{opt} nel caso in cui si è adottata la funzione di fitness f_2, l'andamento oscillatorio risulta significativamente smorzano rispetto al caso precedente, in modo particolare quando si raggiungono valori di fitness alti. Il miglior andamento della distanza sulla fitness lascia intendere che lo spazio di ricerca relativo alla funzione di fitness f_2 sia effettivamente caratterizzato da un numero minore di ottimi locali rispetto al caso precedente, ipotesi confermata anche dal fatto che l'AG mostra una più spiccata tendenza verso l'ottimo globale (si noti la linea di tendenza in figura 5.10b). Il miglior individuo evoluto nei quattro test eseguiti ha ottenuto un valore di fitness decisamente elevato: $f_2 = 0.88$ (figura



(a) (b)

Figura 5.10: Risultati degli esperimenti di ottimizzazione sul caso di studio idealizzato adottando f_1 come funzione di fitness: (a) mostra l'evoluzione temporale della fitness (in nero la fitness del miglior individuo, in grigio la fitness media dell'intera popolazione), ottenuta come media sui quattro esperimenti eseguiti; (b) mostra la distanza tra gli individui evoluti dall'AG e l'ottimo globale P_{opt} (in grigio la linea di tendenza).

5.10a). Rispetto al caso precedente, tale valore è da ritenersi più che soddisfacente, specialmente in considerazione del fatto che la funzione di fitness f_2 risulta dal prodotto di due fattori entrambi a valori nell'intervallo $[0, 1]$.

5.7 Discussione

L'ottimizzazione dei modelli sviluppati nel contesto del metodo empirico proposto da Di Gregorio e Serra [59] costituisce un aspetto estremamente importante nella passaggio dalla modellizzazione alla simulazione dei fenomeni d'interesse. In questa fase, come mostrato nelle prime applicazioni ai modelli S4a ed S4b, gli Algoritmi Genetici possono rappresentare un valido strumento, in particolar modo quando applicati in contesti di calcolo parallelo. Tuttavia, la presenza di vincoli morfologici, che generalmente influenzano la dinamica degli eventi macroscopici considerati, ha messo in luce la delicatezza della fase d'ottimizzazione, evidenziando la possibilità di evolvere soluzioni sub-ottimali con conseguenze estremamente pericolose in vista di possibili applicazioni alla definizione di mappe di rischio in aree ad alta probabilità di catastrofi naturali.

Ci si può convince facilmente che la definizione di una funzione di fitness che tenga in considerazione tutte le proprietà dell'evento reale possa risolvere il problema della convergenza verso un ottimo locale (il numero di ottimi locali dovrebbe

infatti ridursi drasticamente o, addirittura annullarsi), ma tale operazione risulta di fatto impraticabile per la mancanza di dati. Di conseguenza, diventa estremamente importante mediare opportunamente tra l'esigenza di adottare la migliore funzione di fitness possibile e la disponibilità dei dati neccessari alla sua definizione. Un primo tentativo in questo senso è stato proposto, in questo capitolo, definendo due differenti funzioni di fitness per l'ottimizzazione del modello S4b e verificandone l'efficacia su un caso di studio idealizzato in cui era noto l'ottimo globale. Studi di questo tipo possono dare importanti informazioni sull'efficacia della fase di ottimizzazione e, nei casi in cui la migliore funzione di fitness che è possibile definire in relazione ai dati disponibili risenta eccessivamente del problema degli ottimi locali, può suggerire ulteriori investigazioni per condurre a buon fine la calibratura del modello. Possibili soluzioni potrebbero consistere nel verificare il comportamento del modello ottimizzato su casi di studio equivalenti ma che evolvono in contesti morfologici differenti o eseguire la fase di ottimizzazione contemporaneamente su più casi di studio. In entrambi i casi, tuttavia, oltre all'ovvio aumento dei tempi di calcolo, si ripropone il problema della disponibilità dei dati. In definitiva, l'ottimizzazione di modelli di simulazione di fenomeni macroscopici complessi, come quelli presentati in questo e nel precedente capitolo, può essere eseguita efficacemente con Algoritmi Genetici, a patto però di disporre di informazioni sufficienti per la definizione di buone funzioni di fitness o, per lo meno, di un numero sufficiente di casi di studio tra essi equivalenti.

Capitolo 6

Conclusioni

Come si può evincere dal lavoro presentato nei capitoli precedenti, la ricerca oggetto della Tesi si è collocata in un contesto prettamente multidisciplinare, poichè ha richiesto competenze che vanno dall'Informatica e dall'Intelligenza Artificiale, alla Fisica e alla Geologia. In questo contesto, il contributo di chi scrive è stato particolarmente significativo nella fase modellistica e nella fase d'ottimizzazione, grazie anche alle conoscenze e alle competenze acquisite durante il percorso formativo del Dottorato che ha visto negli Automi Cellulari e negli Algoritmi Genetici due importanti argomenti di studio e di ricerca.

Le potenzialità di questi metodi, già note alla comunità scientifica internazionale per i risultati teorici (presentati nei capitoli 2 e 3) e per le numerose applicazioni in settori scientifici e ingegneristici anche molto eterogenei (applicazioni degli Algoritmi Genetici che vanno dalle Scienze Cognitive all'Ottimizzazione sono stati presentati nel capitolo 2; applicazioni degli Automi Cellulari che vanno dalle Vita Artificiale alla Fisica sono stati presentati nel capitolo 3), sono state confermate dai risultati del lavoro discusso in questa Tesi che ha visto l'applicazione degli Automi Cellulari alla modellizzazione e simulazione di fenomeni macroscopici complessi, nello specifico alla simulazione di flussi di detrito rapidi, e la loro ottimizzazione tramite Algoritmi Genetici.

La fase modellistica si è inquadrata nel contesto generale del metodo empirico proposto da Di Gregorio e Serra (presentato nel capitolo 4) per la modellizzazione di fenomeni macroscopici complessi con Automi Cellulari. Partendo da un semplice modello per la simulazione di flussi detritici, SCIDDICA, applicato inizialmente alla simulazione delle frane di Tessina e di Monte Ontake, sono stati sviluppati i nuovi modelli della famiglia "Sx", capaci di riprodurre sempre meglio fenomeni di complessità più elevata come i flussi di detrito rapidi. Buoni risultati sono stati

ottenuti già a partire dalla versione S1, applicata per prima alla simulazione del disastro di Sarno. Comunque, significativi miglioramenti nelle *performance* del modello si sono riscontrate nella versione S3-hex (anch'essa presentata nel capitolo 4) grazie al raffinamento dei processi elementari della funzione di transizione e all'introduzione dello spazio cellulare esagonale in sostituzione dell'originario spazio cellulare a celle quadrate; le applicazioni agli eventi di Sarno del 1998 (discusse ancora nel capitolo 4) hanno fornito, infatti, risultati soddisfacenti su tutti i casi di studio analizzati.

Gli sviluppi successivi hanno visto un primo tentativo di considerare, in maniera esplicita, gli effetti inerziali dei flussi detritici rapidi che hanno interessato l'area di Sarno. Così sono state sviluppate due nuove versioni del modello SCIDDICA, la versione S4a e la versione S4b (presentate nel capitolo 5), ognuna delle quali è caratterizzata da un approccio specifico al problema. Entrambe le versioni, comunque, possono essere considerate estensioni del precedente modello S3-hex. Il tentativo di considerare gli effetti inerziali nelle versioni S4a ed S4b ha, tuttavia, accentuato il problema della calibratura dei due modelli, cioè la determinazione dei valori dei parametri tali da consentire una riproduzione soddisfacente degli eventi reali, richiedendo l'adozione di un metodo d'ottimizzazione automatica. In entrambi i casi gli Algoritmi Genetici sono stati preferiti ad altri metodi d'ottimizzazione, sia per la loro natura di algoritmi di ricerca "general purpose", e quindi applicabili anche in contesti in cui non esistono tecniche d'ottimizzazione standard, sia perchè, come accennato in precedenza, essi sono stati oggetto di studio del corso di Dottorato.

Gli Algoritmi Genetici sono stati inizialmente applicati all'ottimizzazione della versione S4a sul caso di studio di Curti. I risultati sono stati estremamente soddisfacenti e l'applicazione del modello ottimizzato ha prodotto la migliore simulazione mai ottenuta sullo stesso caso di studio, sia in termini qualitativi che quantitativi. Tuttavia la fase d'ottimizzazione del modello S4a, eseguita in un ambiente di calcolo sequenziale, ha richiesto tempi di calcolo estremamente elevati (dell'ordine dei mesi) suggerendo l'adozione di un ambiente di Calcolo Parallelo per la successiva ottimizzazione del modello S4b. A tal fine è stato possibile considerare una soluzione di tipo Cluster Beowulf, macchina parallela composta, nel presente caso specifico, da 16 processori di classe Pentium IV a 1.4 GHz interconnessi tramite una comune rete LAN a 100 Mbps. L'applicazione degli Algoritmi Genetici Paralleli di tipo Master-Slave, che rappresentano una mera parallelizzazione dell'Algoritmo Genetico sequenziale, alla versione S4b sullo stesso caso di studio di Curti, ha consentito di ridurre i significativamente i tempi d'esecuzione. Alcune prove

(descritte nel capitolo 5) hanno mostrato, infatti, la scalabilità lineare del sistema; in altri termini, il tempo d'esecuzione diminuisce proporzionalmente al numero di processori impiegati.

L'abbattimento dei tempi d'esecuzione, conseguente all'adozione dell'ambiente di calcolo parallelo, ha consentito, inoltre, un primo studio qualitativo sulla dinamica dell'Algoritmo Genetico, utile per comprendere l'efficacia del metodo d'ottimizzazione in un settore applicativo in cui non esiste un'esperienza che ne possa confermare l'attendibilità. A tal fine, si è ipotizzato un caso di studio ideale in cui fosse nota la soluzione ottimale del problema di ricerca e si è analizzata la dinamica dell'Algoritmo Genetico in relazione a due differenti funzioni di fitness. La prima, f_1, considerava esclusivamente il confronto areale tra l'evento simulato e l'evento reale, la seconda, f_2, teneva in considerazione anche gli effetti dell'erosione del suolo lungo il percorso del flusso detritico reale e simulato. I risultati di questo studio hanno evidenziato come la ricerca della soluzione ottimale possa essere significativamente influenzata dalla scelta della funzione di fitness. Nel caso di f_1 la dinamica dell'Algoritmo Genetico è risultata in un comportamento essenzialmente oscillatorio, indicando la presenza di numerosi ottimi locali nello spazio di ricerca; nel caso di f_2, invece, l'andamento oscillatorio, caratteristico della prima funzione di fitness, si è ridotto sensibilmente, mostrando inoltre una spiccata tendenza a convergere verso l'ottimo globale. Tuttavia, in entrambi i casi l'Algoritmo Genetico ha evoluto buone soluzioni ma, mentre nel caso di f_2 questo può essere considerato un buon risultato che può confermare l'efficacia del metodo, nel caso di f_1 non è possibile sostenere una tale affermazione per l'alta probabilità che la convergenza si abbia verso una soluzione non ottimale che, peraltro, può risultare anche significativamente "distante" da quella ottimale.

Appare abbastanza intuitivo convincersi del fatto che più informazioni (estensione areale, erosione, ecc.) siano considerate nella funzione di fitness, più alta risulti l'affidabilità del processo di ottimizzazione. Tuttavia, spesso la disponibilità dei dati relativi agli eventi discussi in questa tesi si limita alla sola estensione areale e, di conseguenza può risultare non banale definire buone funzioni di fitness per l'Algoritmo Genetico. In ogni caso, i risultati di questo lavoro suggeriscono, anche in considerazione del fatto che si è in presenza di un settore d'applicazione nuovo per gli Algoritmi Genetici, l'adozione di una fase preliminare d'indagine in cui l'attendibilità del metodo sia verificata su casi di studio idealizzati come quello discusso nel capitolo 5 e, in caso di fallimento, un'ulteriore fase d'investigazione che possa confermare i risultati ottenuti su differenti casi di studio, equivalenti dal punto di vista geomorfologico a quello scelto per la fase di ottimizzazione vera e

propria.

La ricerca presentata e discussa in questo lavoro non può essere, tuttavia, considerata definitiva. I modelli discussi nel capitolo precedente rappresentano, infatti, solo il primo tentativo di considerare gli effetti della quantità di moto, e quindi di esplicitare il ruolo della velocità, nel contesto locale dell'Automa Cellulare e, in modo particolare, nell'ambito specifico del metodo empirico proposto da Di Gregorio e Serra. In secondo luogo appare fondamentale proseguire l'indagine sull'efficacia, e quindi sull'opportunita d'applicazione, degli Algoritmi Genetici (o eventualmente altri metodi alternativi) all'ottimizzazione di modelli ad Automi Cellulari per la simulazione di fenomeni macroscopici complessi; qualora se ne possa definitivamente stabilire l'efficacia, infatti, il metodo potrebbe essere facilmente esteso, con presumibili benefici, a differenti modelli che s'inquadrano nello stesso contesto.

Ringraziamenti

E' doveroso, in conclusione, rivolgere i più sentiti ringraziamenti alle seguenti persone:

al Dottor Davide Marocco per aver condiviso con me il (talvolta non semplicissimo) percorso di studi del Dottorato,

alla Professoressa Eleonora Bilotta e al Professor Pietro Pantano per i preziosi consigli e per l'autonomia concessami nella ricerca,

al Professor Angelo Cangelosi per aver supervisionato la parte della Tesi relativa agli Algoritmi Genetici,

al Dottor William Spataro per aver pazientemente rivisto le bozze del manoscritto alla ricerca (fruttuosa) delle sviste in esse contenute,

al Dottor Giulio Iovine per aver supervisionato gli aspetti prettamente geologici del lavoro,

alla Dottoressa Valeria Lupiano per l'elaborazione di alcune immagini in ambiente GIS.

Bibliografia

[1] D. Ackley and M. Littman. Interaction between learning and evolution. In C.G. Langton, C. Taylor, J.D. Farmer, and S. Rasmussen, editors, *Artificial Life II*. Addison-Wesley, 1992.

[2] E. Alba and M. Tomassini. Parallelism and evolutionary algorithms. *IEEE Transactions on Evolutionary Computation*, 6:443–462, 2002.

[3] J. Alcaraz and C. Maroto. A robust genetic algorithm for resource allocation in project scheduling. *Annals of Operation Research*, 102:83–109, 2001.

[4] A. Armanini. On the dynamic impact of debris flows. In A. Armanini and F. Michiue, editors, *Recent development on debris*, volume 64 of *LNES*, pages 208–226. Springer Verlag, Berlin, 1997.

[5] A. Armanini and L. Fraccarollo. Critical conditions for debris flows. In C.I. Chen, editor, *Debris-flow hazard mitigation: mechanics, prediction, and assessment*, pages 434–443, 1997.

[6] D. Ashlock, M. Smucker, E.A. Stanley, and L. Tesfatsion. Preferential partner selection in an evolutionary study of prisoner's dilemma. *BioSystems*, 37:99–125, 1996.

[7] M.V. Avolio, D. D'Ambrosio, S. DiGregorio, R. Fata, G. Iovine, V. Lupiano, R. Rongo, and W. Spataro. A cellular automata model for different complexity landslides. In G. Nardi, editor, *Proceedings GIAST*, pages 101–114, Sansepolcro, Arezzo, Italy, 1999.

[8] M.V. Avolio, D. D'Ambrosio, S. DiGregorio, R. Fata, R. Rongo, and W. Spataro. SCIDDICA: an incremental cellular model for landslides of increasing complexity. In *Proceedings 25th European Geophysical Society*

General Assembly. Geophysical Research Abstracts (in CD), volume 2, page 146, Nice, France, April 2000.

[9] M.V. Avolio, S. DiGregorio, F. Mantovani, A. Pasuto, R. Rongo, S. Silvano, and W. Spataro. Simulation of the 1992 Tessina landslide by a cellular automata model and future hazard scenarios. *International Journal of Applied Earth Observation and Geoinformation*, 2:41–50, 2000.

[10] M.V. Avolio, S. DiGregorio, R. Rongo, M. Sorriso Valvo, and W. Spataro. Hexagonal cellular automata model for debris flow simulation. In A. Buccianti, G. Nardi, and R. Potenza, editors, *Proceedings International Association for Mathematical Geology Conference*, pages 920–924, Naples, 1998. Litografia Editrice.

[11] R. Axelrod. *The evolution of cooperation*. Basic Books, New York, 1984.

[12] R. Axelrod. The evolution of strategies in the iterated prisoner's dilemma. In L. Davis, editor, *Genetic Algorithms and Simulated Annealing*, pages 32–41. Morgan Kaufmann Publisher, Inc., 1988.

[13] R. Axelrod and D. Dion. The further evolution of cooperation. *Science*, 242(4884):1385–1390, 1988.

[14] I. Azpeitia and J. Ibáñez. Spontaneous emergence of robust cellular replicators. In S. Bandini, B. Chopard, and M. Tomassini, editors, *Cellular Automata. Proceedings 5th International Conference on Cellular Automata for Research and Industry, ACRI 2002*, LNCS, pages 32–43, Geneve, Switzerland, October 2002. Spinger, Berlin.

[15] J.E. Baker. Adaptive selections methods for genetic algorithms. In J.J. Grefenstette, editor, *Proceedings of the First Conference on Genetic Algorithms and Their Applications*. Erlbaum, 1985.

[16] J.E. Baldwin. A new factor in evolution. *American Naturalist*, 30:441–451, 536–553, 1896.

[17] D. Barca, G.M. Crisci, S. DiGregorio, and F.P. Nicoletta. Cellular automata for simulating lava flows: a method and examples of the etnean eruptions. *Transport Theory and Statistical Physics*, 23:195–232, 1994.

[18] R.D. Beer and J.C. Gallagher. Evolving dynamical neural networks for adaptive behavior. *Adaptive Behavior*, 1:91–122, 1992.

[19] R.K. Belew. Evolution, learning, and culture: Computational metaphores for adaptive algorithms. *Complex Systems*, 4:11–49, 1990.

[20] E. Berlekamp, J.H. Conway, and R. Guy. *Winning for Your Mathematical Plays*. Academic Press, New York, 1982.

[21] M. Bernaschi, S. Succi, and H. Chen. Accelerated lattice Boltzmann schemes for steady-state flow simulations. *Journal of Scientific Computing*, 16(2):135–144, 2001.

[22] E. Bilotta, A. Lafusa, and P. Pantano. Is self-replication an embedded characteristic of artificial/living matter? In Standish, Abbass, and Bedau, editors, *Proceedings Artificial Life VIII*. MIT Press, 2002.

[23] T. Blickle and L. Theile. A mathematical analysis of tournament selection. In L.J. Eshelman, editor, *Proceedings of the Sixth International Conference on Genetic Algorithms*, pages 9–16, Hillsdale, New Jersey, 1995. Lawrence Erlbaum Associates.

[24] E.N. Bromhead. Pore water pressure manipulation in computerised slope stability analysis. In *Proceedings Midland Geotechnical Society Conference on Computer Applications in Geotechnical Engineering*, pages 175–184, Birmingham, 1986.

[25] E. Cantù-Paz. *Efficient and Accurate Parallel Genetic Algorithms*. Kluwer Academic Publishers, 2000.

[26] L. Cascini and P. Versace. Eventi pluviometrici e movimenti franosi. In *Atti del XVI Convegno Nazionale di Geotecnica*, volume 3, pages 171–184, 1986.

[27] H. Chatè and P. Manneville. Criticality in cellular automata. *Physica D*, (45):122–135, 1990.

[28] S. Chatterjee, C. Carrera, and L.A. Lynch. Genetic algorithms and travelling salesman problem. *European Journal of Operational Research*, 93:490–510, 1996.

[29] H. Chen, S. Chen, and W.H. Matthaeus. Recovery of Navier-Stokes equations using lattice-gas Boltzmann method. *Physical Review A*, 45:5339, 5342 1992.

[30] B. Chopard and M. Droz. *Cellular Automata Modeling of Physical Systems*. Cambridge University Press, 1998.

[31] B. Chopard, P. Luthi, A. Masselot, and A. Dupuis. Cellular automata and lattice Boltzmann techniques: An approach to model and simulate complex systems. *Advances in Complex System*, 5(2):103–246, 2002.

[32] B. Chopard and A. Masselot. Cellular automata and lattice Boltzmann methods: a new approach to computational fluid dynamics and particle transport. *Future Generation Computer Systems*, 16:249–257, 1999.

[33] H.H. Chou and J.A. Reggia. Emergence of self-replicating structures in cellular automata space. *Physica D*, 110:252–276, 1997.

[34] S. Clerici and S. Perego. Simulation of the parma river blockage by the corniglio landslide (northern italy). *Geomorphology*, 33:1–23, 2000.

[35] E.F. Codd. *Cellular Automata*. Academic Press, New York, NY, 1968.

[36] P. Cortez, M. Rocha, and J. Neves. A lamarckian approach for neural network training. *Neural Processing Letters*, 15:105–116, 2002.

[37] G. Crisci, S. DiGregorio, and G.A. Ranieri. A cellular space model of basaltic lava flow. In *Proceedings International Conference on Applied Modelling and Simulation*, volume 11, pages 65–67, Paris, France, 1982.

[38] G.M. Crisci, A. DiFrancia, S. DiGregorio, R. Rongo, and W. Spataro. An improved cellular automaton model for lava flow simulation. In S.J. Lippard, A. Næss, and R. Sinding-Larsen, editors, *Proceedings International Association for Mathematical Geology Conference*, pages 317–323, Trondheim, Norway, 1999.

[39] G.M. Crisci, S. DiGregorio, O. Pindaro, and S.A. Ranieri. Lava flow simulation by a discrete cellular model: first implementation. *International Journal of Modelling and Simulation*, 6:137–140, 1986.

[40] G.M. Crisci, S. DiGregorio, R. Rongo, M. Scarpelli, W. Spataro, and S. Calvari. Revisiting the 1669 Etnean eruptive crisis using a cellular automata model and implications for volcanic hazard in the Catania area. *Journal of Volcanology and Geothermal Research*, (123):211–230, 2003.

[41] G.M. Crisci, S. DiGregorio, R. Rongo, and W. Spataro. Lava area hazard determination for the town of Nicolosi (Sicily) by cellular automata. In A. Buccianti, G. Nardi, and R. Potenza, editors, *Proceedings International Association for Mathematical Geology Conference*, pages 920–924, Naples, 1998. Litografia Editrice.

[42] P.H Crowley, L. Provencher, S. Sloane, L.A. Dugatkin, B. Sphon, L. Rogers, and M. Alfieri. Evolving cooperation: the role of individual recognition. *BioSystems*, 37:49–66, 1996.

[43] J.P. Crutchfield. The calculi of emergence: Computation, dynamics, and induction. *Physica D*, 75:11–54, 1994.

[44] J.P. Crutchfield and J.E. Hanson. Attractor vicinity decay for a cellular automaton. *CHAOS*, 3(2):215–224, 1993.

[45] J.P. Crutchfield and M. Mitchell. The evolution of emergent computation. In *Proceedings of the National Academy of Sciences*, number 23, USA 92, November 1995.

[46] J.P. Crutchfield, M. Mitchell, and R. Das. The evolutionary design of collective computation in cellular automata. In J.P. Crutchfield and P.K. Schuster, editors, *Evolutionary Dynamics-Exploring the Interplay of Selection, Neutrality, Accident, and Function*, New York, 2002. Oxford University Press.

[47] D. D'Ambrosio, S. DiGregorio, S. Gabriele, and R. Gaudio. A cellular automata model for soil erosion by water. *Physics and Chemistry of the Earth – Part B*, 26(1):33–40, 2001.

[48] D. D'Ambrosio, S. DiGregorio, S. Gabriele, and R. Gaudio. Un metodo di trasformazione di reticoli a maglia quadrata in reticoli a maglia esagonale e viceversa. Technical Report 585, CNR-IRPI - Sezione di Cosenza, Maggio 2002.

[49] D. D'Ambrosio, S. DiGregorio, and G. Iovine. Simulating debris flows through a hexagonal cellular automata model: Sciddica S3-hex. *Natural Hazards and Earth System Sciences*, 3:545–559, 2003.

[50] D. D'Ambrosio, S. DiGregorio, G. Iovine, V. Lupiano, L. Merenda, R. Rongo, and W. Spataro. Simulating the Curti-Sarno debris flow through cellular automata: the model SCIDDICA (release S2). *Physics and Chemistry of the Earth*, 27:1577–1585, 2002.

[51] D. D'Ambrosio, S. DiGregorio, G. Iovine, V. Lupiano, R. Rongo, and W. Spataro. Fisrt simulations of the Sarno debris flows through cellular automata modelling. *Geomorphology*, 54:91–117, 2003.

[52] D. D'Ambrosio, W. Spataro, and G. Iovine. Parallel genetic algorithms for optimising cellular automata models of natural complex phenomena: an application to debris-flows. *Computer & Geosciences*, submitted.

[53] K. DeJong, D.B. Fogel, and H.P. Schwefel. A history of evolutionary computation. In T. Back, D.B. Fogel, Z. Michalewicz, and T. Baeck, editors, *Handbook of Evolutionary Computation*. IOP Publishing Ltd and Oxford University Press, 1997.

[54] K.A. DeJong. *Analysis of Behavior of a Class of Genetic Adaptive Systems*. Phd thesis, University of Michigan, 1975.

[55] M. DelPrete, F.M. Guadagno, and A.B. Hawkins. Preliminary report on the landslides of 5 May 1998, Campania, Southern Italy. *Bulletin of Engineering Geology and the Environment*, 57:113–129, 1998.

[56] G.M. Crisci A. DiFrancia, S. DiGregorio, F.P. Nicoletta, R. Rongo, and W. Spataro. SCIARA.2: a cellualar automata model for lava flow simulation. In V.P. Glahan, editor, *Proceedings International Association for Mathematical Geology Conference*, pages 11–16, Barcelona, 1997. Addendum.

[57] S. DiGregorio. L'applicazione del modello SCIARA all simulazione dell'eruzione etnea del 2002. Comunicazione personale, 2003.

[58] S. DiGregorio, R. Rongo, C. Siciliano, M. Sorriso Valvo, and W. Spataro. Mount Ontake landslide simulation by the cellular automata model SCIDDICA-3. *Physics and Chemistry of the Earth – Part A*, 24(2):131–137, 1999.

[59] S. DiGregorio and R. Serra. An empirical method for modelling and simulating some complex macroscopic phenomena by cellular automata. *Future Generation Computer Systems*, 16:259–271, 1999.

[60] S. DiGregorio, R. Serra, and M. Villani. Applying cellular automata to complex environmental problems: The simulation of the bioremediation of contaminated soil. *Theoretical Computer Science*, 217:131–156, 1999.

[61] S. DiGregorio and G. Trautteur. On reversibility in cellular automata. *Journal of Computer and System Science*, 11(3):382–391, 1975.

[62] G. Doolen, editor. *lattice Gas Method for Partial Differential Equations*. Addison-Wesley, 1990.

[63] G.H. Eisbacher and J.J. Clague. *Destructive mass movement in high mountains: hazard and management*, chapter Geological Survey of Canada, pages 84–16. Ottawa, Canada, 1984.

[64] D. Floreano and F. Mondada. Evolution of homing navigation in a real mobile robot. *IEEE Transactions on Systems, Man, and Cybernetics – Part B: Cybernetics*, 26(3):396–407, 1996.

[65] L.J. Fogel, A.J. Owens, and M.J. Walsh. *Artificial Intelligence through Simulated Evolution*. Wiley, 1966.

[66] E. Fredkin and T. Toffoli. Conservative logic. *International Journal of Theoretical Physics*, 21:219–53, 1982.

[67] R.M. French and A. Messinger. Genes, phenes, and the baldwin effect: Learning and evolution in a simulated population. In R.A. Brooks and P. Maes, editors, *Artificial Life IV*. MIT Press, 1994.

[68] U. Frish, D. d'Humières, and P. Lallemand. Lattice gas models for 3D hydrodynamics. *Europhysics Letters*, 291, 1987.

[69] U. Frish, B. Hasslacher, and Y. Pomeau. Lattice gas automata for the Navier-Stokes equation. *Physical Review Letters*, 56(14):1505–1508, 1986.

[70] M. Gardner. Mathematical games: The fantastic combinations of John Conway's new solitaire game.

[71] C. Ghezzi and D. Mandrioli. *Informatica teorica*. Clup, Milano, 1989.

[72] D.E. Goldberg. *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison-Wesley, 1989.

[73] D.E. Goldberg. A note on Boltzmann tournament selection for genetic algorithms and population-oriented simulated annealing. *Complex Systems*, 4:445–460, 1990.

[74] T. Gomi and K. Ide. Emergence of gaits of a legged robot by collaboration through evolution. In P.K. Simpson, editor, *IEEE World Congress on Computational Intelligence (WCCI'98)*. IEEE Press., 1998.

[75] W. Gropp, E. Lusk, and A. Skjellum. *Using MPI: Portable Parallel Programming with the Message Passing Interface (Scientific and Engineering Computation)*. MIT Press, Cambridge, Massachusetts, 2nd edition, 1999.

[76] H.A. Gutowitz. A hierarchical classification of cellular automata. *Physica D*, (45):136–156, 1990.

[77] J.E. Hanson. *Computational Mechanics of Cellular Automata*. PhD thesis, University of California, Berkeley, 1993.

[78] J.E. Hanson and J.P. Crutchfield. The attractor-basin portrait of a cellular automaton. *Journal of Statistical Physics*, 66:1415, 1992.

[79] J.E. Hanson and J.P. Crutchfield. Computational mechnics of cellular automata: An example. *Physica D*, 103:169–189, 1997.

[80] J. Hardy, Y. Pomeau, and G. de Pazzis. Thermodynamics and hydrodynamics for a modeled fluid. *Journal of Mathematical Physics A*, 13(5):1949–1961, 1976.

[81] P.G. Harrald and D.B. Fogel. Evolving continous beaviours in the iterated prisoner's dilemma. *BioSystems*, 37:135–145, 1996.

[82] I. Harvey. The puzzle of the persistent question mark: A case study of genetic drift. In S. Forrest, editor, *Proceedings of the Fifth International Conference on Genetic Algorithms*. Morgan Kaufmann, 1993.

[83] F. Higuera and J. Jimenez. Boltzmann approach to lattice gas simulations. *Europhysics Letters*, 9(7):663–668, 1989.

[84] C.E. Hinton and S.J. Nowlan. How learning can guide evolution. *Complex Systems*, 1:495–502, 1987.

[85] A. Hoffman. The ecology of cooperation. *Theory and Decision*, 50:101–118, 2001.

[86] J.H. Holland. Nonlinear environments permitting efficient adaptation. *Computer and Information Sciences II (New York: Academic)*, 1967.

[87] J.H. Holland. *Adaptation in Natural and Artificial Systems*. University of Michigan Press, Ann Arbor, 1975.

[88] G. Iovine, D. D'Ambrosio, and S. DiGregorio. Applying genetic algorithms for calibrating a hexagonal cellular automata model for the simulation of debris flows characterised by strong inertial effects. *Geomorphology*, in press.

[89] G. Iovine and S. DiGregorio. Sciddica (s2): alcune riflessioni sull'applicazione di un modello ad automi cellulari per la simulazione di colate detritiche. *Bollettino della Società Geologica Italiana*, 122:63–84, 2003.

[90] G. Iovine, S. DiGregorio, D. D'Ambrosio, and V. Lupiano. Debris flows and cellular automata: an example of simulation from the 1998 disaster of Sarno (Italy). In *Geomorphology: from expert opinion to modelling*, pages 55–64, Strasbourg, France, April 2002. SODIMPAL Imprimeur, Rouen.

[91] G. Iovine, S. DiGregorio, and V. Lupiano. Debris-flow susceptibility assessment through cellular automata modeling: an example from 15-16 December 1999 disaster at Cervinara and San Martino Valle Caudina (Campania, southern Italy). *Natural Hazards and Earth System Sciences*, 3:457–468, 2003.

[92] R.M. Iverson. The physics of debris flows. *Reviews in Geophysics*, 35:245–296, 1997.

[93] R.M. Iverson and R.P. Denlinger. Flow of variably fluidized granular masses across three-dimensional terrain. *Journal of Geophysical Research*, 106(B1):537–566, 2001.

[94] R.M. Iverson, R.P. Denlinger, R.G. LaHusen, and M. Logan. Two-phase debris-flow accross 3-D terrain: Model predictions and experimental tests.

In G.F. Wieczorek and N.D. Naeser, editors, *Debris-Flow Hazard Mitigation: Mechanics Prediction, and Assessment. Proceedings 2nd International Conference on Debris Flow Hazards Mitigation*, pages 521–529. Balkema, Rotterdam, 2000.

[95] N. Jakobi. Running across the reality gap: Octopod locomotion evolved in a minimal simulation. In P. Husbands and J.A. Meyer, editors, *Evolutionary Robotics. First European Workshop*, pages 39–58, Berlin, 1998. Springer-Verlag.

[96] F. Jiménez-Morales. Evolving three-dimensional cellular automata to perfom a quasiperiod-3(p3) collective behavior task. *Physical Review E*, 60(4):4934–4940, 1999.

[97] F. Jiménez-Morales. An evolutionary approach to the study of non-trivial collective behavior in cellular automata. In S. Bandini, B. Chopard, and M. Tomassini, editors, *Cellular Automata. Proceedings 5th International Conference on Cellular Automata for Research and Industry, ACRI 2002*, LNCS, pages 32–43, Geneve, Switzerland, October 2002. Spinger, Berlin.

[98] F. Jiménez-Morales and J.J. Luque. Collective beavior of a probabilistic cellular automaton with two absorbing phases. *Physical Letters A*, 181:33–38, 1993.

[99] A.M. Johnson. *Physical Processes in Geology.* Freeman and Cooper, San Francisco, 1970.

[100] J. Kari. Reversibility of 2d cellular automata is undecidable. *Physica D*, 45:379–385, 1990.

[101] D.K. Keefer, R.C. Wilson, R.K. Mark, E.E. Brabb, W.M. Brown III, S.D. Ellen, E.L. Harp, G.F. Wieczorek, C.S. Alger, and R.S. Zatkin. Real-time landslide warning during heavy rainfall. *Science*, 238:921–925, 1987.

[102] J.E. Kesseli. Disintegrating soil slips of the coast ranges of central california. *Journal of Geology*, 51(5):342–352, 1943.

[103] V.I. Klenov. 2-d debris-flow simulation. In G.F. Wieczorek and N.D. Naeser, editors, *Debris-Flow Hazards Mitigation: Mechanics, Prediction, and Assessment. Proceedings 2nd International Conference on Debris-Flow Hazards Mitigation*, pages 547–550, Taipei, Taiwan, 2000. A.A. Balkema, Rotterdam.

[104] J.R. Koza. *Genetic Programming: On the Programming of Computers by Means of Natural Selection*. MIT Press, 1992.

[105] K.W.C. Ku and M.W. Mak. Exploring the effects of lamarckian and baldwinian learning in evolving recurrent neural networks. In *Proceedings of the IEEE International Conference on Evolutionary Computation*, pages 617–621, 1997.

[106] D. Laigle and L. Marchi. Example of mud/debris flow hazard assessment, using numerical models. In G.F. Wieczorek and N.D. Naeser, editors, *Debris-flow hazards mitigation: Mechanics, prediction, and assessment. Proceedings 2nd International Conference on debris-flow hazard mitigation*, pages 417–424, Taipei, Taiwan, 2000.

[107] L. Landau. *Fisica dei Fluidi*. Edizioni MIR, 1984.

[108] C.G. Langton. Self-reproduction in cellular automata. *Physica D*, (10):135–144, 1984.

[109] C.G. Langton. Studying artificial life with cellular automata. *Physica D*, (22):120–149, 1986.

[110] C.G. Langton. *Computation at the edge of caos*. PhD thesis, University of Michigan, 1990.

[111] C.G. Langton. Computation at the edge of caos: phase transition and emergent computation. *Physica D*, (42):12–37, 1990.

[112] C.G. Langton. *Artificial Life*, volume IV of *Santa Fe Institute Studies in the Sciences of Complexity*, pages 1–47. Addison-Wesley, Reading, MA, 1998.

[113] P. Larrañaga, C.M.H. Kuijpers, R.H. Murga, I. Inza, and S. Dizdarevic. Genetic algorithms for the travelling salesman problem: A review of representation and operators. *Artificial Intelligenece Review*, 13:129–170, 1999.

[114] A.A. Lipnitskii. Use of genetic algorithms for solution of the rectangle packing problem. *Cybernetics and Systems Analysis*, 38(6):943–946, 2002.

[115] A.L. Little and V.E. Price. The use of an electronic computer for slope stability computation. *Géotechnique*, 8:113–120, 1958.

[116] K.F. Liu and K.W. Lai. Numerical simulation of two-dimensional debris flows. In G.F. Wieczorek and N.D. Naeser, editors, *Debris-flow hazards mitigation: Mechanics, prediction, and assessment. Proceedings 2nd International Conference on debris-flow hazard mitigation*, pages 531–535, Taipei, Taiwan, 2000.

[117] L.S. Luo. Theory of the lattice Boltzmann method: Lattice Boltzmann models for nonideal gases. *Physical Review E*, 62(4):4982–4996, 2000.

[118] B.D. Malamud and D.L. Turcotte. Self-organized criticality applied to natural hazards. *Natural Hazards*, 20:93–116, 1999.

[119] B.D. Malamud and D.L. Turcotte. Cellular automata models applied to natural hazards. *IEEE Computing in Science & Engineering*, 2(3):42–51, 2000.

[120] E. Marchi and A. Rubatta. *Meccanica dei fluidi. Principi e applicazioni.* UTET, Torino, 1981.

[121] R.K. Mark and S.D. Ellen. Statistical and simulation models for mapping debris-flow hazard. In A. Carrara and F. Guzzetti, editors, *Geographical Information Systems in assessing natural hazards*, pages 93–106, 1995.

[122] D. Marocco, A. Cangelosi, and S. Nolfi. The role of social and cognitive abilities in the emergence of communication: Experiments in evolutionary robotics. In *EPSRC/BBSRC International Workshop Biologically-Inspired Robotics HP*, pages 174–181, 2002.

[123] A.R. McBirney and T. Murase. Rheological properties of magmas. *Annual Review on Earth and Planetary Sciences*, 12:337–357, 1984.

[124] H. McIntosh. Wolfram's class IV automata and a good life. *Physica D*, (45):105–121, 1990.

[125] G.R. McNamara and G. Zanetti. Use of the Boltzmann equation to simulate lattice-gas automata. *Physical Review Letters*, 61:2332–2335, 1988.

[126] M. Mitchell. *An Introduction to Genetic Algorithms.* MIT Press, 1996.

[127] H. Miyamoto and S. Sasaki. Simulating lava flows by an improved cellular automata method. *Computers & Geosciences*, 23:283–292, 1997.

[128] E. Moore. Machine models of self-reproduztion. In *Proceedings Symposium on Applied Mathematics*, pages 17–33, 1962.

[129] N.R. Morgenstern and V.E. Price. A numerical method for solving the equations of stability of general slip surfaces. *Computer Journal*, 9:388–393, 1967.

[130] S. Munroe and A. Cangelosi. Learning and the evolution of language: the role of cultural variation and learning cost in the baldwin effect. *Artificial Life*, 8:311–339, 2003.

[131] A.B. Murray and C. Paola. A cellular model of braided rivers. *Nature*, 371:54–57, 1994.

[132] A.B. Murray and C. Paola. Properties of a cellular braided-stream model. *Earth Surface Processes and Landforms*, 22(11):1001–1025, 1997.

[133] J. Myhill. The converse of Moore's Garden-of-Eden theorem. In *Proceedings Symposium on Applied Mathematics*, pages 658–686, 1963.

[134] United Nations. *Mudflows. Experience and lessons learned from the management of major disasters.* Departiment of Humanitarian Affairs, Geneva, 1987.

[135] S. Nolfi. Evolving non-trivial behaviors on real robots: A garbage collecting robot. *Robotics and Autonomous System*, 22:187–198, 1997.

[136] S. Nolfi. How learning and evolution interact: The case of a learning task which differs from the evolutionary task. *Adaptive Behavior*, 7(2):231–236, 1999 (copiright 2000).

[137] S. Nolfi, J.L. Elman, and D. Parisi. Learning and evolution in neural networks. *Adaptive Behavior*, 3(1):5–28, 1994.

[138] S. Nolfi and D. Floreano. *Evolutionary Robotics: The Biology, Intelligence, and Technology of Self-Organizing Machines (Intelligent Robotics and Autonomous Agents).* MIT Press, 2000.

[139] S. Nolfi and D. Marocco. Evolving visually-guided robots able to discriminate between different landmarks. In J.A. Meyer, A. Berthoz, D. Floreano, H.L.

Roitblat, and S.W. Wilson, editors, *From Animals to Animats 6. Proceedings of the VI International Conference on Simulation of Adaptive Behavior*, Cambridge, MA, 2000. MIT Press.

[140] S. Nolfi and D. Marocco. Evolving robots able to integrate sensory-motor information over time. *Theory in Bioscience*, 120:287–310, 2001.

[141] S. Nolfi and D. Marocco. Active perception: A sensorimotor account of object categorization. In *From Animals to Animats 7. Proceedings of the 7th Intlernational Conference on Simulation of Adaptive Behavior*, 2002.

[142] S. Nolfi and D. Marocco. Evolving robots able to visually discriminate between objects with different size. *International Journal of Robotics and Automation*, 17(4):163–170, 2002.

[143] J.C. Oh. Promoting cooperation using kin biased conditional strategy in the iterated prisoner's dilemma game. *Information Science*, 133:149–164, 2001.

[144] M. Oliphant. The dilemma of saussurean communication. *BioSystems*, 37:31–38, 1996.

[145] G.C. Onwubolu and M. Mutingi. A genetic algorithm approach for cutting stock problem. *Journal of Intelligent Manufacturing*, 14:209–218, 2003.

[146] P. Pacheco. *Parallel Programming With MPI.* Morgan Kaufmann, San Francisco, 1996.

[147] P. Pantano. Autoreplicazione in automi cellulari caratterizzati da regole di transizione non complesse. Comunicazione personale, 2003.

[148] D. Parisi and S. Nolfi. How learning can influence evolution within a non-lamarckian framework. In R. K. Belew and M. Mitchell, editors, *Adaptive Individuals in Evolving Populations*, volume XXVI of *SFI Studies in the Science of Complexity*. Addison-Wesley, 1996.

[149] A. Prügel-Bennett. Finite population effects for ranking and tournament selection. *Complex Systems*, 12(2):183–205, 2000.

[150] A. Prügel-Bennett and J.L. Shapiro. An analysis of genetic algorithms using statistical mechanics. *Physical Review Letters*, 72(9):1305–1309, 1994.

[151] A. Prügel-Bennett and J.L. Shapiro. The dynamics of a genetic algorithm for simple random ising systems. *Physica D*, 104:75–114, 1997.

[152] Y.H. Qian, D. d'Humières, and P. Lallemand. Lattice BGK models for Navier-Stokes equation. *Europhysics Letters*, 17(6):470–484, 1992.

[153] A. Quarteroni and A. Valli. *Numerical Approximation of Partial Differential Equations*. Springer Series in Computational Mathemetics. Springer Verlag, 2nd edition, 1997.

[154] I. Rechenberg. *Evolutionsstrategie: Optinierung Technischer Systeme nach Prinzipien der Biologischen Evolution*. Fromman-Holzboog (Struttgart), 1973.

[155] A. Rogers and A. Prügel-Bennett. The dynamics of a genetic algorithm on a model hard optimization problem. *Complex Systems*, 11(6):437–464, 2000.

[156] A. Roli and F. Zambonelli. Emergence of macro spatial structures in dissipative cellular automata. In S. Bandini, B. Chopard, and M. Tomassini, editors, *Cellular Automata. Proceedings 5th International Conference on Cellular Automata for Research and Industry, ACRI 2002*, LNCS, pages 32–43, Geneve, Switzerland, October 2002. Spinger, Berlin.

[157] D.H. Rothman and S. Zaleski. Lattice-gas models of phase separation: interfaces, phase transitions, and multiphase flow. *Review of Modern Physics*, 66(4):1417–1479, 1994.

[158] H. Rouse. *Engineering Hydraulics*. John Wiley & Sons, Chichester, 1950.

[159] T.P. Runarsson and M.T. Jonsson. Genetic production systems for intelligent problem solving. *Journal of Intelligent Manufacturing*, 10:181–186, 1999.

[160] F. Sandersen, S. Bakkeoi, E. Hestnes, and K. Lied. The influence of meteorological factors on the initiation of debris flows, rockfalls, rockslides and rockmass stability. In K. Senneset, editor, *Landslides, Proceedings 7th International Symposium on Landslides*, volume 1, pages 94–114, Trondheim, 1996. Balkema, Rotterdam.

[161] K. Sassa. Motion of landslides and debris flows. Report for grant-in-aid for scientific research, (project no.61480062), Japanese Ministry on Education, Science and Culture, Tokyo, 1988.

[162] K. Sassa. Special lecture: Geotechnical model for the motion of landslides. In C. Bonnard, editor, *Landslides. Proceedings 5th International Symposium on Landslides*, volume 1, pages 37–56, Lausanne, Switzerland, 1988. A.A. Balkema, Rotterdam.

[163] E. Segre and C. Deangeli. Cellular automaton for realistic modelling of landslides. *Nonlinear Processes in Geophysics*, 2(1):1–15, 1995.

[164] G.A. Sena, D. Magherbi, and G. Isern. Implementation of a parallel genetic algorithm on a cluster of workstations: Travelling salesman probmen, a case study. *Future Generation Computer Systems*, 17:477–488, 2001.

[165] R. Serra. *Calcolo Parallelo, automi cellulari e modelli per sistemi complessi*, chapter Prefazione, pages 11–14. Franco Angeli, 1999.

[166] C.C. Simpson. The baldwin effect. *Evolution*, 7:110–117, 1953.

[167] R. Smith. The application of cellular automata to the erosion of landforms. *Earth Surface Processes and Landforms*, 16:273–281, 1991.

[168] S. Succi. *Automi Cellulari*. Francoangeli, Milano, 1991.

[169] S. Succi. Lattice Boltzmann schemes for quantum applications. *Computer Physics Communications*, 146:317–323, 2002.

[170] T. Takahashi. Initiation and flow of various types of debris flow. In G.F. Wieczorek and N.D. Naeser, editors, *Debris-flow hazards mitigation: Mechanics, prediction, and assessment*, Proceedings 2nd International Conference on debris-flow hazard mitigation, pages 15–25, Taipei, Taiwan, August 2000.

[171] J.W. Thacher. Universality in the von neumann cellular model. In A.W. Burks, editor, *Essays on Cellular Automata*, pages 103–131, Champaign, 1970. University of Illinois Press.

[172] T. Toffoli. Cellular automata as an alternative to (rather than an approximation of) differential equations in modeling physics. *Physica D*, 10:117–127, 1984.

[173] T. Toffoli and N. Margolus. *Cellular Automata Machines*. MIT Press, Cambridge, 1987.

[174] T. Toffoli and N. Margolus. Invertible cellular automata: A review. *Physica D*, 45:229–253, 1990.

[175] M. Tomassini. Parallel and distributed evolutionary algorithms: A review. In P. Neittaanmäki k. Miettinen, M. Mäkelä and J. Periaux, editors, *Evolutionary Algorithms in Engineering and Computer Scince*, pages 113–133, Chichester, UK, 1999. John Wiley & Son.

[176] M. Tomassini and M. Venzi. Artificially evolved asynchronous cellular automata for the density task. In S. Bandini, B. Chopard, and M. Tomassini, editors, *Cellular Automata. Proceedings 5th International Conference on Cellular Automata for Research and Industry, ACRI 2002*, LNCS, pages 32–43, Geneve, Switzerland, October 2002. Spinger, Berlin.

[177] E.F. Toro. *Riemann solvers and numerical methods for fluid dynamics*. Springer, 1997.

[178] M. Villani, M. Padovani, M. Andretta, M. Mazzanti, R. Serra, B. Mueller, H.P. Ratzke, R. Rongo R., W. Spataro, and S. DiGregorio. Bioremediation modeling: from the pilot plant to the field. In V.S. Magar, T.M. Vogel, C.M. Aelion, and A. Leeson, editors, *Innovative Methods in support of Bioremediation. In situ Bioremediation of Petroleum Hydrocarbon and Other Organic Compounds, The Sixth International In situ and On-Site Bioremediation Symposium*, San Diego , California, 2001. Battelle Press, Columbus, Richland.

[179] J. von Neumann. *Theory of self-reproducing automata*. University of Illinois Press, Urbana, Illinois, 1966. (Edited and completed by A. Burks).

[180] C.H. Waddington. Canalization of development and the inheritance of acquired characters. *Nature*, 150:563–565, 1942.

[181] J.R. Weimar. *Simulation with Cellular Automata*. Logos Verlag, Berlin, 1997.

[182] D. Whitley, V. Gordan, and K. Mathias. Lamarkian evolution, the baldwin effect and function optimization. In Y. Davidor et al., editor, *Parallel Solving From Nature*, pages 6–15. Springer, 1994.

[183] G.F. Wieczorek. Effect of rainfall intensity and duration on debris flows in central Santa **C**ruz Mountains, California. *Reviews in Engineering Geology*, 7:93–104, 1987.

[184] G.F. Wieczorek and N.D. Naeser, editors. *Debris-flow hazards mitigation: mechanics, prediction, and assessment*, Rotterdam, 2000. Proceedings 2nd International Conference on Debris Flow Hazards Mitigation, Balkema.

[185] R.V. Withman and W.A. Bailey. Use of computers for slope stability analysis. *Journal of the Soil Mechanics and Foundations Division*, 93:475–498, 1967.

[186] S. Wolfram. Statistical mechanics of cellular automata. *Reviews of Modern Physics*, 55:601–644, 1983.

[187] S. Wolfram. Cellular automaton fluids 1: Basics theory. *Journal of Statistical Physics*, 45(3/4):471–526, 1986.

[188] S. Wolfram. *A new kind of Science*. Wolfram Media Inc., Champaign, 2002.

[189] A. Wuensche. Classifying cellular automata automatically; finding gliders, filtering, and relating space-time patterns, attractor basins, and the z parameter. *COMPLEXITY*, 4(3):47–66, 1999.

[190] A. Wuensche. Basins of attraction in cellular automata; order-complexity-chaos in small universes. *COMPLEXITY*, 5(6):19–26, 2000.

[191] A. Wuensche and M. Lesser. *The Global Dynamics of Cellular Automata: An Atlas of Basin of Attraction Fields of One-Dimensional Cellular Automata*. Addison-Wesley, 1992.

Elenco delle figure

Elenco delle tabelle