SAPIENZA
UNIVERSITÀ DI ROMA

Complex Adaptive Systems
Modelli Computazionali

Facoltà di Ingegneria Informatica, Informatica e Statistica
Dipartimento di Informatica
Corso di laurea in Informatica

Pietro Dondi
Matricola 1797080

Responsabile
Pietro Cenciarelli

A.A. 2020-2021

Indice

Glossario

CS = complex systems (sistemi complessi)

CAS = complex adaptive systems (sistemi complessi adattivi)

CASS = complex adaptive social systems (sistemi complessi adattivi e sociali)

0. Introduzione

In natura siamo circondati da sistemi complessi, noi stessi siamo un sistema complesso, uno dei più studiati e misteriosi, a sua volta composto da molti sottosistemi complessi. Ad essi dobbiamo la nostra vita e la nostra evoluzione nel tempo. È un sistema complesso il nostro sistema nervoso e soprattutto il nostro encefalo, lo sono i sistemi di neural networks di deep learning a cui si sono ispirati, lo è l'universo globale e ogni sua galassia, lo è la città in cui viviamo, lo è il clima e tutto ciò che influenza il riscaldamento globale, lo è la diffusione di un virus, lo sono molti software sviluppati e lo sono tantissimi altri sistemi e realtà a cui siamo abituati, le cui complessità e profondità diamo per scontate.
La disciplina della Teoria della Complessità, che si occupa di studiare, catalogare e modellare sistemi complessi, è giovane e sta compiendo i primi passi verso la comprensione di questo affascinante argomento. Uno dei problemi che si riscontrano, ma che al contempo attraggono di più, è l'interdisciplinarità della materia, la sua continua evoluzione e adattabilità ad ogni nuovo sistema che emerge o muore influenzato da altri sistemi complessi. Potremmo affermare che la teoria della complessità sia essa stessa un sistema complesso.
La Teoria della Complessità si occupa di una vasta gamma di sistemi complessi molto differenti gli uni dagli altri, ognuno dei quali presenta spesso peculiarità e caratteristiche univoche. Si sono delineati tuttavia, nel tempo, gruppi di sistemi complessi accomunati da proprietà condivise dai sistemi dello stesso gruppo. Il sottogruppo a cui dedicheremo maggiormente la nostra attenzione è quello dei sistemi complessi adattivi (Complex Adaptive Systems – CAS).
L'obiettivo di questo lavoro è comprendere la natura dei sistemi complessi adattivi e modellarne uno tramite modelli computazionali.
Per raggiungere questo scopo è necessario svolgere prima dei passi introduttivi e teorici che forniranno le linee guida al modello. Serve quindi capire cosa sia un CAS, che caratteristiche abbia, cosa andiamo a modellare, come verifichiamo che il modello è conforme alla descrizione, cosa è necessario alla modellazione. Per far questo la teoria è essenziale sia come guida che come supporto. Tuttavia, è bene notare come non esista una teoria centrale e globalmente condivisa per sistemi complessi adattivi. Il primo vero obiettivo sarà allora quello di provare a far convergere il più possibile la teoria sviluppata negli anni in un unico lavoro. Questo è reso possibile solo con il contributo di vari libri e articoli sull'argomento scritti dai più celebri ricercatori di CAS, i quali consigliano loro stessi di provare a costruire una teoria sulla base di varie fonti, essendo l'argomento fortemente ampio e multidisciplinare.
Prima di saltare direttamente a descrivere i CAS è importante prima contestualizzarli nella Teoria dei Sistemi, la disciplina che si occupa dello studio dei sistemi in generale.

Dato il contesto sarà allora possibile capire meglio l'eredità che hanno i CAS nella Teoria dei Sistemi e, più specificatamente, nella Teoria della Complessità.
Si vorrà successivamente estrapolare delle caratteristiche chiave comuni ai sistemi complessi. In seguito, si potrà finalmente specializzarsi sui sistemi complessi adattivi.
Una volta che si è arrivati a poter restringere il campo di studi, l'attenzione ricadrà sul provare a dare una definizione di adattabilità, la parola chiave di tutto il lavoro. Quindi risulterà necessario, per completare la teoria, studiare tutte le caratteristiche tipiche di CAS.
Questa tesi vuole proporsi come un compendio di studi dei Sistemi Complessi Adattivi, sulla base dei lavori svolti da numerosi ricercatori. Questa disciplina è ancora molto aperta, giovane, poco studiata, altamente multidisciplinare, fortemente dinamica ed in evoluzione. Le fonti di questo lavoro vengono quindi da molteplici risorse. Le principali sono Emergence di Steven Johnson ed un libro e due articoli tutti omonimi tra loro: Complex Adaptive Systems, scritti rispettivamente da Miller e Page, John Holland e Serena Chan. Tutte le fonti sono comunque elencate nelle sezioni di Bibliografia e Sitografia. Non è semplice fondere in un'unica trattazione un argomento così poco strutturato e vasto, sulla base di ricerche interdisciplinari e indipendenti tra loro. Tuttavia, la necessità di creare una base teorica è risultata necessaria e favorevole alla comprensione profonda dell'argomento, delle sue tecniche di modellazione e delle sue implicazioni pratiche.

1. Introduzione alla Teoria dei Sistemi

Come anticipato nell'introduzione, è bene fare una contestualizzazione dei sistemi complessi adattivi nella disciplina della Teoria dei Sistemi che si occupa, tra le altre cose, anche di sistemi complessi. Si ritiene essenziale fare un breve excursus su altre categorie di sistemi. Non verranno definiti né elencati tutti i sistemi possibili. La disciplina è veramente così vasta da portarci velocemente fuori dagli obiettivi di questa tesi. L'attenzione verrà riposta solo sui tipi di sistemi che hanno influenzato maggiormente la Teoria della Complessità, sotto disciplina della Teoria dei Sistemi.

I sistemi complessi adattivi sono sistemi, aperti, dinamici, non lineari, complessi, adattivi.

In quest'ordine, che è dal generale al particolare, da un insieme ai suoi sotto insiemi, daremo una definizione di questi tipi di sistemi

Partendo dalla definizione di sistema, si scenderà continuamente in sottosistemi e sottogruppi meno generali per arrivare a studiare e collocare in modo chiaro i sistemi complessi. Studieremo allora vari sistemi, quelli aperti, quelli dinamici, quindi quelli non lineari, quindi quelli complessi, caotici ed infine quegli adattivi, il nostro obiettivo finale.

Nel paragrafo "Una visione d'insieme" viene rappresentata graficamente, e corredata di esempi, questa digressione progressiva dei sistemi trattati.

I sistemi

Un sistema è un *insieme di elementi connessi fra loro*. Tali connessioni sono caratterizzate da relazioni di causa ed effetto. Un evento nel sistema causa uno o più eventi, i quali effetti si ripercuotono nel sistema o nel suo ambiente. Per ambiente del sistema intendiamo tutto ciò che non fa parte del sistema ed è quindi un fattore esogeno ma con il quale il sistema può interagire.

I sistemi chiusi e aperti

Si differiscono subito due macro categorie di sistemi: aperti e chiusi. Un sistema è aperto se interagisce con il suo ambiente. Le modalità di interazione sono molteplici, ad esempio lo sono gli scambi di materia, di energia, di elementi immateriali, di comunicazioni, di valori o parametri, di obiettivi. Un sistema è chiuso se è isolato dal suo ambiente, non interagisce con esso.

L'essere umano è un sistema aperto, una pianta, un virus, ma anche il web o una pentola d'acqua. Allo stato attuale della fisica si considera un sistema chiuso l'universo, non conoscendo ancora la presenza di un ambiente esterno ad esso. Altri esempi possibili sono pressoché infiniti.

I sistemi statici e dinamici

Un sistema si definisce dinamico se almeno uno dei suoi elementi o connessioni cambia nel tempo. Vengono rappresentati da *stati* che cambiano in funzione del tempo. Sono quindi sistemi che mutano nel tempo e la loro evoluzione è modellata in stati definiti. Un esempio elementare di sistema dinamico è il moto di un punto nello spazio.

Al contrario un sistema si definisce statico se nessun suo elemento o connessione cambia nel tempo. Ovvero se esso può essere rappresentato sempre nello stesso stato.

Un sistema dinamico manifesta delle transizioni da uno stato all'altro nel corso della sua vita ed evoluzione nel tempo. È possibile che si trovi in uno stesso stato per un certo intervallo di tempo, ed essere quindi statico, per poi subire una nuova transizione verso un nuovo stato. Spesso vengono modellati tramite equazioni differenziali che ne descrivono l'andamento nel tempo.

Esistono molteplici sottogruppi dei sistemi dinamici differiti per la loro relazione con il fattore tempo, ad esempio se esso è discreto o continuo e se il sistema è variante o invariante rispetto ad esso. Tuttavia, ignoriamo questa differenziazione perché non contribuisce veramente a caratterizzare i sistemi complessi. Due importanti sottogruppi dei sistemi dinamici che invece ci serve conoscere sono i sistemi dinamici lineari e non lineari.

I sistemi lineari e non lineari

Un sistema lineare è un sistema descritto da funzioni lineari che rispettano il *principio di sovrapposizione*, secondo il quale un sistema lineare reagisce a sollecitazioni o input diversi in modo indipendente da essi. La somma degli effetti nel sistema è la somma degli effetti dei singoli input.

Formalmente sia H un sistema dinamico lineare e $\alpha_1 x_1 + \cdots + \alpha_n x_n$ una combinazione lineare di input x_i linearmente indipendenti al sistema, ognuno moltiplicato da uno scalare $\alpha_i \in \mathbb{R}$. Il sistema risponde ad ognuno di essi in modo indipendente ovvero la risposta del sistema all'insieme di tutti gli input è data dalla somma delle singole risposte ad ogni variabile, che per omogeneità vale $H(\alpha x_i) = \alpha H(x_i)$, e per additività: $H(\alpha_1 x_1 + \cdots + \alpha_n x_n) = \alpha_1 H(x_1) + \cdots + \alpha_n H(x_n)$.

Un sistema dinamico lineare viene descritto da un insieme di equazioni per le quali l'obiettivo principale è studiarne almeno una soluzione, che esiste sempre, dato lo stato iniziale e le funzioni lineari di transizione di stato. Inoltre, un sistema lineare ha la proprietà chiave di produrre sempre lo stesso output dato lo stesso input.

Essi sono fondamentali perché molti sistemi non lineari vengono spesso semplificati ed il loro comportamento approssimato tramite un processo di *"linearizzazione"* che consiste nell'approssimare il sistema non lineare nei suoi punti di equilibrio tramite l'uso di funzioni lineari.

Un sistema dinamico non lineare invece è un sistema in cui la risposta in output non è proporzionale agli input ricevuti. Essi sono descritti da funzioni non lineari, in genere equazioni differenziali e funzioni polinomiali di grado *maggiore* ad uno. Non obbediscono al principio di sovrapposizione di sistemi lineari; infatti, il loro comportamento non è indipendente dall'insieme degli input bensì altamente *dipendente* da essi e quindi spesso *imprevedibile*, e talvolta quindi anche contro intuitivo.

Ciò è dovuto alla proprietà chiave del sistema di essere composto da sottosistemi od elementi, le quali relazioni rispettano principi di causa effetto non lineari. Questo implica che un piccolo cambiamento di stato di uno di questi componenti tipicamente porta a cambiamenti a cascata esponenziali su tutti gli altri componenti del sistema in modo imprevedibile e talvolta caotico. Ancor più particolare è la differenza di comportamento del sistema dati gli stessi input, infatti, l'output atteso potrà non essere lo stesso o essere unico e irripetibile. Per sistemi dinamici non lineari non è sempre possibile trovare una soluzione, essa può non esistere o non è possibile trovarla con i mezzi che abbiamo attualmente a disposizione. Più importante di trovare la soluzione al sistema è infatti capire come esso evolve nel tempo, come reagisce e si comporta a stimoli differenti, come si diffonde un cambiamento al suo interno e la più importante di tutte: il sistema tende ad un *equilibrio*, ad uno stato *ottimale*?

I sistemi complessi

Si occupa di studiare, classificare e modellare sistemi complessi la Teoria della Complessità, una branca della Teoria dei Sistemi che si occupa di aspetti così vasti che abbracciano molteplici campi della scienza, dalla matematica alla fisica, dalla chimica alla biologia, dall'informatica alle scienze naturali, dalla politica all'economia e alle scienze sociali. È difficile trovare una qualsiasi scienza priva di sistemi complessi.

Non esiste una definizione di sistema complesso. Almeno non esiste una definizione *formale*, *unica* e comunemente *accettata* dalla comunità scientifica. Questo primo grande scoglio nel voler comprendere i sistemi complessi per modellarne uno non ha una vera e propria soluzione. Il metodo usato per approcciare la descrizione, mai definizione, di sistemi complessi è quella di elencare e approfondire tutte le loro più

note proprietà e delinearne un insieme minimale senza le quali non si può parlare di complessità.

I sistemi complessi (CS da qui in poi, da Complex Systems) sono un sottoinsieme dei sistemi dinamici non lineari. Possono essere chiusi ma più spesso sono sistemi aperti. Il loro nome deriva dall'essere sistemi composti da numerose variabili dipendenti e connesse e quindi rappresentano sistemi molto difficili da risolvere con i soliti mezzi, e talvolta non hanno proprio una soluzione.

Un sistema complesso è un sistema composto da un insieme di sottosistemi, o elementi o agenti (continueremo sempre ad usare tutti e tre i termini), con la proprietà chiave che essi siano interconnessi tra loro, a formare un complesso insieme di collegamenti. Tali connessioni tra un elemento e l'altro del sistema creano una rete di connessioni che genera uno o più comportamenti *globali* del sistema (per differenziarlo dalle connessioni *locali* tra componenti) che si chiama *comportamento "emergente*". Le connessioni di causa-effetto tra agenti del sistema sono molto accentuate in sistemi complessi, spesso difficili da comprendere e prevedere. Tutto ciò contribuisce alla complessità del sistema.

In natura la quasi totalità di CS è composta da elementi interdipendenti tra di loro. Ognuno influenza anche in minima parte ogni altro e questo rende impossibile prevedere con certezza il comportamento del sistema nel tempo e nello spazio.

Data la natura dinamica, non-lineare, interconnessa, interdipendente e a comportamento emergente di tali sistemi, il classico approccio alla modellazione, di semplificare e ridurre il sistema nei suoi componenti alla base fallisce nel suo intento. Ci occuperemo più avanti di modellazione ma è bene sapere fin da subito come l'approccio da perseguire migliore sia di tipo *olistico,* ovvero studiando il sistema "in toto" o "dall'alto" senza scomporlo. La regola aurea dei CS è infatti "*la somma dei componenti non è uguale al tutto*".

Tale approccio ha trovato nell'informatica, e quindi in modelli computazionali supportati da modelli matematici, il suo ambiente naturale di studio e modellazione. I calcolatori si pongono in modo naturale come ambiente adatto a modellare sistemi complessi data la loro potenza computazionale che nel tempo si è sempre più evoluta e potenziata, permettendo un progressivo miglioramento nella comprensione della natura dei sistemi complessi.

Il termine complesso fa riferimento anche alla composizione del sistema di un numero cospicuo di elementi interagenti e dipendenti, la quale modellazione – del sistema - per vie matematiche o computazionali è per l'appunto complesso. In informatica un sistema complesso implica spesso la presenza di almeno un algoritmo che li modella ad alta complessità computazionale.

La maggioranza dei CS è *non deterministico*, non esiste in generale una funzione che lega tutti gli elementi del sistema o tutte le sue connessioni. In virtù di non

determinismo non è possibile determinare con precisione matematica il comportamento, a lungo termine, del sistema. Anche conoscendo tutti i possibili dati sullo stato del sistema, le funzioni di evoluzione e cambiamento di ogni sua componente, è possibile fare previsioni stocastiche sul suo stato successivo ma non c'è determinismo matematico.

Esistono anche CS *deterministici* composti tipicamente da pochi agenti, le cui regole di connessione sono ben determinate. Tuttavia, sono accomunati ai CS non deterministici dalla proprietà di comportarsi in modo imprevedibile, lontana da possibili previsioni e tendenti, avvolte, a manifestare comportamenti caotici.

I sistemi dinamici di ogni tipo sono caratterizzati dalla capacità di tendere verso *stati ottimali* e verso *punti di equilibrio* che rendono il sistema stabile e soggetto a minimi scambi di energia con il suo ambiente. I CS, loro sottoinsieme, manifestano anch'essi la presenza di punti di equilibrio, e di condizioni di stabilità. Nonostante ciò, essi manifestano più frequentemente comportamenti dovuti da condizioni di *instabilità* e *disequilibrio*. In natura, infatti, un CS tipicamente cambia continuamente e si sposta da uno stato di equilibrio all'altro, oppure ci si avvicina il più possibile. Raramente si trovano CS che si stabilizzano in modo perpetuo su un punto di equilibrio e, se presentano punti di equilibrio sono spesso più di uno, inoltre tali punti sono difficilmente costanti nel tempo.

La trattazione di questo argomento in profondità, ovvero della presenza e tendenza dei sistemi dinamici in generale verso punti di equilibrio, esula dagli argomenti di questa tesi essendo specifici di altri ambiti di ricerca. Nonostante ciò, analizzeremo più avanti il punto di vista dei sistemi complessi adattivi e vedremo come questi ultimi differiscono profondamente dal concetto di equilibrio, di ottimalità e di stabilità.

Tutti i sistemi complessi condividono un minimo insieme di peculiarità comuni. Tuttavia, per natura, ognuno di essi è unico e distinguibile da ogni altro. È però possibile, e necessario, delineare delle divisioni in sottogruppi individuando delle caratteristiche comuni per ciascuna categoria. Questo anche per concentrare la nostra attenzione verso un solo gruppo alla volta migliorandone la comprensione, lo studio e quindi la sua modellazione.

Successivamente vediamo due importanti sottogruppi dei sistemi complessi: i sistemi caotici e i sistemi complessi adattivi.

I sistemi caotici

Un sistema caotico è un sistema dinamico non lineare e complesso, caratterizzato da tre proprietà essenziali: una forte *sensibilità* (di tipo tipicamente esponenziale) agli input che riceve, esibire transitività topologica, avere un insieme denso di orbite periodiche.

Non entreremo nel dettaglio di tutte le caratteristiche che definiscono un sistema caotico. Bisognerebbe avventurarsi nel mondo della topologia, degli attrattori, della periodicità, che sono ambiti trattati in corsi avanzati di matematica e fisica ed esulano dall'argomento della tesi. È però importante sapere per esaustività che esiste un forte legame tra la teoria dei sistemi complessi e la teoria del caos.

L'unica precisazione che intendiamo fare è cosa si intende per *caos*. Caos, in teoria dei sistemi, non vuol dire casuale o aleatorio, anche se talvolta un comportamento caotico presenta un certo grado di aleatorietà, ma è inteso come forte sensibilità agli stimoli o input che riceve internamente o dal suo ambiente. Tale caratteristica è intrinsecamente legata ad un comportamento non prevedibile e non lineare. Il legame fortemente dipendente e sensibile tra input e output del sistema e tra componenti stessi viene definito come caotico. Un sistema può definirsi caotico anche con pochissimi elementi costituenti, le quali interazioni altamente non lineari e, feedback loop prevalentemente positivi, tendono a portare il sistema in stati sempre più caotici. Un esempio tipico è il *doppio pendolo* in fisica.

Quattro possibili fasi di un doppio pendolo. Due pendoli uniti fra loro, una estremità di uno dei due è fissata mentre l'altra è mobile ed unita al secondo pendolo. Le linee di sfondo sono create immaginando di disegnare su un foglio i movimenti dell'estremità della seconda parte libera del pendolo.

Un aspetto importante da notare di questo esempio, coerente con quanto descritto sinora: il sistema (composto da un doppio pendolo immerso tipicamente in un ambiente chiuso e isolato) si comporta sempre in modo diverso ogni volta che viene lasciato libero di muoversi dalla sua posizione iniziale (la prima figura a sinistra). Le variazioni possono essere minime o numerose, comunque sia, date le stesse condizioni iniziali, il sistema si comporta sempre in modo diverso e imprevedibile, quindi caotico. Se poi interagisce col sistema, ad esempio toccando brevemente o dando un input di forza al pendolo, esso reagisce in modo ogni volta differente, producendo effetti visibili a cascata registrati dal tratto di disegno.

Contrariamente a quanto si pensi, esistono sistemi caotici *deterministici*, ovvero governati da funzioni e leggi note che legano le variabili o gli elementi del sistema. Nonostante il determinismo, ciò che li rende caotici è la difficoltà crescente nel tempo di essere accurati con possibili calcoli. Il sistema a lungo andare differisce sempre più dalle previsioni iniziali diventando quindi non prevedibile il comportamento a lungo termine e spesso contro intuitivo.

Introduzione ai sistemi complessi adattivi

I primi studi sui CS risalgono tra la fine dell'Ottocento e i primi anni del Novecento soprattutto nel campo fisico e matematico, ma per molto tempo non fu mai istituita una disciplina od un modus operandi comune tra gli scienziati che approcciavano questa nuova disciplina.

Nella metà del secolo scorso iniziarono a confluire conoscenze multidisciplinari che contribuirono a creare una Teoria della Complessità. Teoria che è tuttora in sviluppo e revisione continua.

Per quanto riguarda lo studio in particolare di sistemi complessi adattivi, fu solo negli anni 80 che un gruppo copioso di scienziati, professori, ricercatori e studenti di diverse discipline, si riunì negli Stati Uniti presso l'*Istituto di Santa Fe*, New Mexico, per delineare le fondamenta dello studio dei sistemi complessi adattivi contribuendo enormemente a rafforzare la Teoria della Complessità. Tra queste menti brillanti e visionarie, di cui alcuni futuri premi Nobel, citiamo John Holland, Murray Gell-Mann, Philip Anderson, John Miller, George Cowan.

La loro eredità viene tuttora raccolta da altri scienziati che nel tempo si sono dedicati a questi studi. Con orgoglio ricordiamo il premio Nobel conferito nel 2021 a Giorgio Parisi, professore di Fisica dell'università La Sapienza di Roma per il suo studio sui sistemi complessi nell'ambito dei cambiamenti climatici.

In questa tesi focalizzeremo la nostra attenzione, come già accennato, su una macrocategoria di CS: i *Sistemi Complessi Adattivi* (Complex Adaptive Systems – CAS, da qui in poi).

Definiamo adattabilità, informalmente, la capacità di un sistema complesso di reagire agli stimoli esterni o interni al sistema, di raccogliere informazioni e feedback, stimolando un processo di apprendimento e provocando nel sistema un'evoluzione e quindi una nuova fase di adattamento. Nei prossimi capitoli torneremo su questa definizione e ne approfondiremo il significato.

I CAS compongono la stragrande maggioranza di CS presenti in natura. La maggior parte della letteratura scientifica si concentra sui CAS essendo il sottogruppo più numeroso e affascinante dei CS.

La distinzione che adottiamo per distinguere CS da CAS è informale. Il motivo risiede fondamentalmente nella mancanza di convergenze verso una definizione comunemente accettata di adattabilità nonché di una categorizzazione unica di CAS. Capita per l'appunto che molti CS vengano descritti come CAS in alcuni articoli o libri, mentre altri non li considerino CAS affatto.

Quasi tutti i CS che rientrano nella nostra definizione di adattabilità hanno in comune una fondamentale componente primaria: la *vita*. Inseriamo quindi nel gruppo dei CAS tutti quei sistemi complessi vivi, o i quali elementi sono a loro volta sottosistemi vivi. Purtroppo, non è possibile nemmeno dare una definizione formale di vita dato che non ne è stata data ancora una che sia globalmente accettata dalla comunità scientifica. Se volessimo tentare di avvicinarci ad un minimo di formalità si potrebbe dire che, generalmente, un sistema vivo è considerato un sistema biologico termodinamico aperto che, tra le altre cose, manifesta *omeostasi* ed *autopoiesi*. Ma ovviamente sappiamo che è molto più di così. A proposito del "*molto più di così*", ritroveremo questa perifrasi quando faremo riferimento alle proprietà o ai comportamenti *emergenti* di un CS. Si è concordi, infatti, che la vita in un sistema biologico complesso è un fenomeno emergente del sistema.

Diamo adesso una descrizione generale di come è fatto un CAS lasciando al capitolo successivo l'onere di descrivere in profondità ogni aspetto.

Un CAS è un sistema, quindi è composto da un insieme di agenti legati da rapporti di causa-effetto. È dinamico, quindi i suoi agenti costituenti e le sue connessioni cambiano nel tempo, tale cambiamento può manifestarsi nei modi più disparati: dalla semplice morte e nascita di nuovi agenti, dalla creazione o cessazione o modifica delle relazioni fra di essi, o in molteplici altri modi. È tipicamente un sistema aperto nel quale l'ambiente gioca un ruolo chiave nel corso della vita di un CAS. È un sistema non lineare, quindi le relazioni di causa-effetto tra i suoi agenti si descrivono per mezzo di funzioni non lineari. È complesso e molta della sua complessità è data dalla sua composizione. Infatti è un sistema in cui sono presenti un gran numero di agenti legati da una fitta rete di connessioni che provoca comportamenti emergenti imprevedibili, difficili da calcolare a priori, talvolta caotici e a propagazione non lineare. Altra componente di complessità è dovuta dall'essere sistemi non deterministici, e dal manifestare causalità verso l'alto e verso il basso contemporaneamente (parleremo di causalità in "Località vs globalità". Infine è adattivo nel senso precedentemente accennato.

Tipicamente un CAS è un sistema vivo, composto da un elevato numero di sotto sistemi o agenti, talvolta essi stessi formano dei sistemi complessi. Ha un ciclo di vita, delle funzioni chiave da svolgere, un comportamento di adattamento al suo ambiente, di continua evoluzione e apprendimento. Manifesta comportamenti emergenti causati da un complesso insieme di eventi, molti dei quali non deterministici, messo in moto dagli agenti interdipendenti e interconnessi del sistema. I comportamenti emergenti che vengono generati si possono manifestare esplicitamente sotto forma di pattern ripetuti o implicitamente spingendo il sistema ad apprendere un comportamento, a reagire ad un evento, ad evolvere al meglio. Vedremo nel prossimo capitolo cosa si intende precisamente con queste affermazioni.

Risulta più facile comprendere questa catena connessa di definizioni e sottosistemi se pensiamo ad un qualsiasi sistema vivo: il nostro corpo ad esempio. Esso è infatti un sistema, dinamico, aperto, non lineare, complesso, adattivo, vivo, composto da un gigantesco numero di agenti e sottosistemi complessi interconnessi ed interdipendenti. Nei prossimi paragrafi verranno presentati ulteriori esempi che chiariranno ancor più quali sono i sistemi che appartengono a questo insieme.

Degni di menzione sono il sottogruppo più celebre dei CAS: i *Sistemi Complessi Adattivi Sociali* (Complex Adaptive Social Systems – CASS, da qui in poi). Un CASS è un CAS i cui agenti (tipicamente dei CAS a loro volta) vivono in *società*, o formano un *gruppo sociale*. La definizione di gruppo sociale deriva direttamente dalla *sociologia* e dalla *sociobiologia.* L'essere umano è un animale fortemente sociale, e lo sono anche moltissime specie di animali come i lupi, i leoni, i buoi, gli elefanti, alcune specie di uccelli, le termiti, le api, le formiche e tante altre specie.

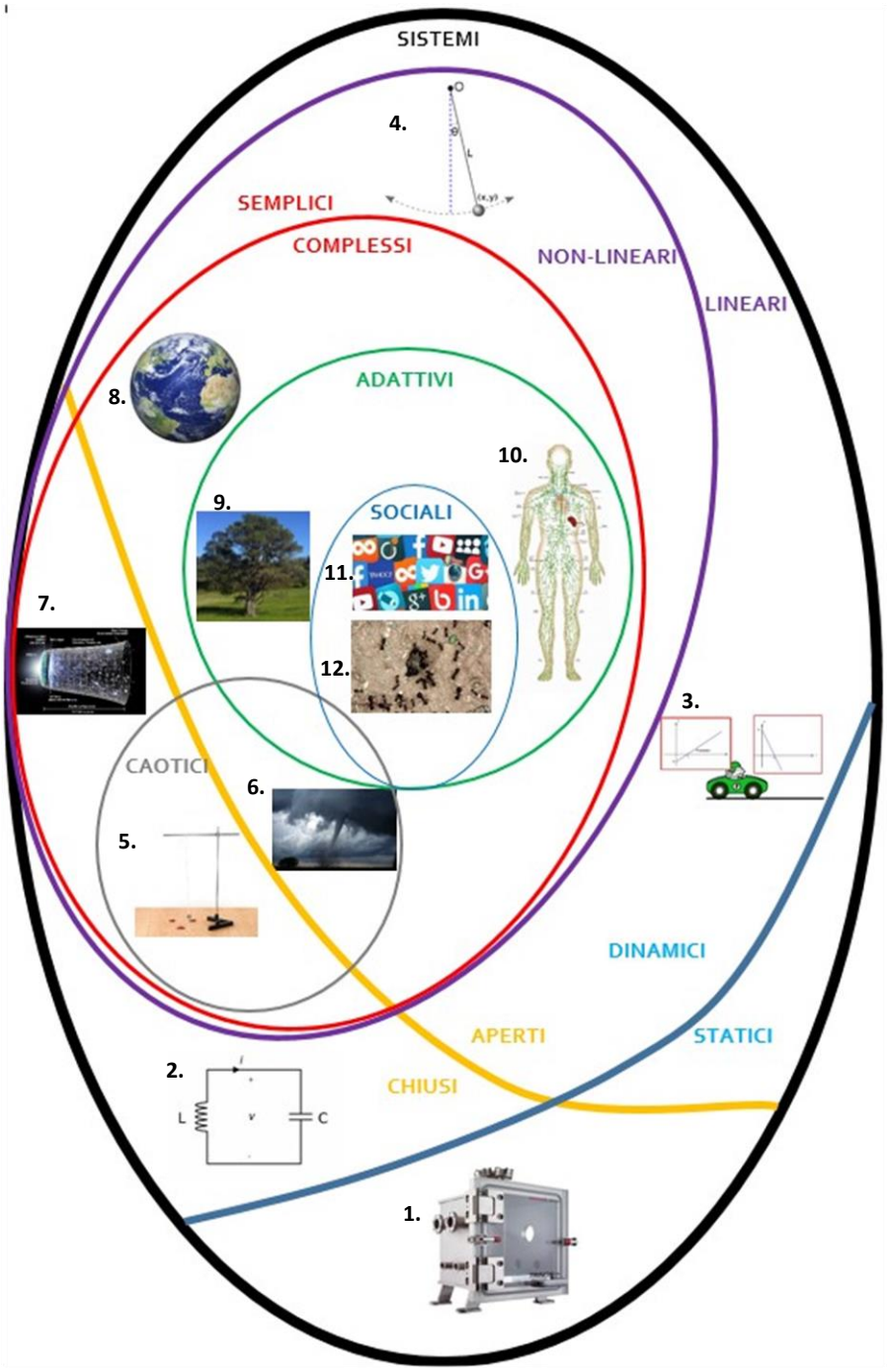
Nella struttura sociale del CASS risiede l'origine della sua complessità. La comunicazione tra agenti sociali è quindi il denominatore comune tra i CASS ma soprattutto è la chiave per comprenderli ed il loro punto focale.

Sono CASS quei sistemi naturali i quali agenti sono sottosistemi complessi adattivi vivi, come un alveare. Sono CASS anche quelli aventi agenti e sottosistemi sia artificiali che naturali. Lo sono un social network, il web in generale o una community su internet, che devono gran parte della loro complessità agli utenti umani (CAS) che li usano formando dei gruppi sociali.

I CASS che andiamo a studiare sono sistemi complessi con forti e distintive note caratteriali di adattamento, composte da un tessuto sociale di elementi che comunicano tra loro, i cui punti chiave sono adattabilità e socialità.

Una visione d'insieme

Di seguito presentiamo una possibile classificazione a diagrammi insiemistici dei sistemi appena descritti. Enfatizziamo la parola *possibile*, dato che non è l'unico modo di classificare i sistemi. In particolar modo la gerarchia organizzata in sottogruppi è corretta e universalmente accettata. Non lo sono i confini dei diagrammi. Quello che viene mostrato è una rappresentazione coerente con la trattazione appena descritta di ogni tipo di sistema ma è bene ricordare che più si scende di gerarchia più i confini tra i sistemi sono correntemente oggetto di dibattito, come ad esempio il confine dei sistemi caotici e di quelli complessi, nonché di quelli adattivi.
Si consiglia di leggere il grafico dall'esterno verso l'interno.



Legenda

1. Camera a vuoto
2. Circuito LC
3. Moto rettilineo uniforme
4. Pendolo semplice
5. Pendolo caotico
6. Uragano
7. Universo
8. Terra
9. Albero
10. Sistema linfatico
11. Social Network
12. Formicaio

Alcuni esempi di sistemi complessi e loro sottogruppi

Elenchiamo successivamente una serie di esempi chiave per capire meglio quali siano i sistemi complessi. L'elenco è diviso in tre sezioni: la prima riguarda esempi relativi a sistemi complessi "puri"; la seconda riguarda sistemi complessi adattivi; la terza riguarda i sistemi complessi adattivi e sociali.

Esempi di CS sono: fenomeni atmosferici come il vento, le tempeste, le nuvole, i tornado, la crosta terrestre, il mare, le onde o la diffusione di un maremoto, il clima di una regione, l'universo, una galassia o cluster di galassie, le molecole di un gas o di un liquido, molti sistemi di fisica atomica e subatomica e molti tipi di reti artificiali.

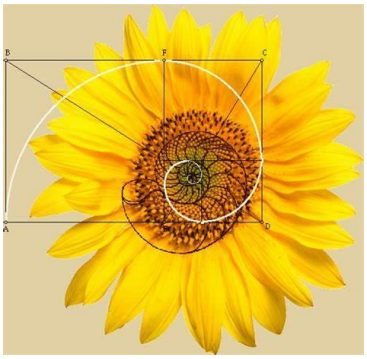
Esempi di CAS sono: un qualsiasi ecosistema, il corpo umano e i suoi sistemi, organi o tessuti (il sistema nervoso, endocrino e linfatico soprattutto), ogni organismo vivente (piante, animali, funghi e batteri) o anche gruppi di organismi viventi eterogenei ed omogenei (non facenti parte di una società), automi cellulari (ad esempio quello descritto dal gioco della vita di Conway), reti neurali di deep learning.

Esempi di CASS sono: un alveare, un formicaio, una mandria di buoi, un branco di lupi, una città, la politica, l'economia e la finanza, un'associazione o un'azienda, un gruppo di persone (famiglie e amici ad esempio), un popolo, il web ed alcune sue realtà come i forum, i social network e le community.

Abbiamo accennato prima come uno dei fenomeni più affascinanti dei CS sia *"l'emergence"*, tradotto informalmente in italiano come "comportamento emergente", caratteristica chiave affrontata nel dettaglio successivamente. Una degli effetti più noti del comportamento emergente di CS è l'affioramento di *pattern*. Di seguito vengono riportati un considerevole numero di pattern o di comportamenti emergenti (la differenza tra i due sarà più chiara quando li definiremo formalmente), secondo la stessa divisione in sezioni di prima.

Esempi di pattern o di comportamenti emergenti da CS: l'innalzamento delle temperature e lo scioglimento dei ghiacciai, la conformazione delle faglie e la distribuzione di terremoti nella crosta terrestre, la nascita di stelle e la formazione di pianeti in una galassia a distanze specifiche dalle stelle, il modo in cui si legano gli atomi di un liquido al passaggio di stato di solidificazione e la sua struttura chimica, la conformazione dei fiocchi di neve, la forma e la composizione

di un terreno naturale, la forma di minerali e gemme, le dune o le righe del vento sulla sabbia, la forma di un'onda.



Esempi di pattern o di comportamenti emergenti da CAS: la vita, la coscienza, l'intelligenza, la forma e distribuzione di solchi sul cervello, la differenziazione e il riconoscimento da parte del sistema linfatico tra cellule interne al sistema e agenti patogeni esterni, la disposizione delle proteine nel capside di un virus, il numero e l'equilibrio di prede, predatori e risorse disponibili in un ecosistema, la conformazione di una pianta o un vegetale o di una singola foglia o petalo (si pensi al numero aureo presente nella conformazione dei broccoli, dei girasoli, delle chiocce o di tante specie di fiori), i periodi di maturazione della frutta o i periodi di germogliazione di un seme, la crescita diffusione e conformazione delle radici e dei rami di un albero, la rete di connessioni di un fungo mucillaginoso (slime mold), in particolare il Physarum polycephalum, la formazione di righe, chiazze, punti sulla pelle di molti animali, la presenza di frattali nelle piante o fiori o sulla pelle di animali, un apprendimento specifico da un sistema di intelligenza artificiale basato su deep learning.

Esempi di pattern o di comportamenti emergenti da CASS: le figure nel cielo che producono gli stormi di uccelli, le fluttuazioni dei prezzi nei mercati finanziari, il traffico in una città, la struttura interna (sale e gallerie) di un formicaio e la sua divisione in caste, la geometria e la termoregolazione di un alveare, la scelta dei prezzi per i prodotti nel mercato, lo sviluppo e la diffusione di un'epidemia, l'evoluzione di una parola o di una lingua in un popolo, la velocità di crescita di una popolazione di organismi viventi, la diffusione di un'idea politica, la diffusione di un coro in uno stadio o una standing ovation in teatro.

2. I Sistemi Complessi Adattivi

Sommario

Siamo finalmente giunti al momento di entrare nel vivo della questione. Analizziamo ora, nel dettaglio, tutte le caratteristiche prima citate nell'introduzione ai sistemi complessi adattivi, e ne affronteremo altre delle quali non si è ancora parlato.

Volendo fare un grande sunto ricordiamo come un CAS sia un sistema, aperto, dinamico, non lineare, complesso, adattivo, vivo, composto da una moltitudine di agenti interconnessi e interdipendenti.

L'obiettivo di questo capitolo è creare una forte base teorica per lo studio, l'analisi di sistemi complessi adattivi. Questa sezione teorica è molto preziosa per poter poi ambire a modellare CAS per vie computazionali ma ancor più per comprendere nel profondo tutto ciò che caratterizza i CAS. La seguente teoria è quindi corredata di definizioni (quando possibile), descrizioni ed esempi che possano il più possibile dare un quadro corretto e completo di questi sistemi complessi.

L'ordine di presentazione delle seguenti caratteristiche è in funzione alla loro priorità. La priorità di un argomento è maggiore se dalla sua definizione o descrizione derivano le definizioni di altre. Esempio: se per parlare di B devo prima parlare di A allora descriveremo prima A in quanto ha priorità più alta.

Nel corso del capitolo useremo la parola "*CS*" quando ci riferiamo a caratteristiche comuni a ogni tipo possibile di sistema complesso; "*CAS*" quando una proprietà è tipica dei sistemi complessi adattivi, sarà quindi la più usata; "*CASS*" quando ci riferiamo a caratteristiche proprie di soli sistemi complessi adattivi e sociali.

Comunque sia il riferimento, ogni CAS presenta tutte le seguenti caratteristiche chiave. Sapremo alla fine di questo elenco, distinguere con chiarezza un CAS da altri tipi di sistemi complessi o non.

Località vs globalità

Useremo continuamente questi due termini quindi è bene specificare il significato che hanno nell'ambito dei CS.

Locale è un aggettivo che associeremo spesso ad un solo agente nel sistema o ad un piccolo numero di agenti. Sono locali i suoi cambiamenti, oppure le sue comunicazioni, connessioni con i suoi agenti vicini. Locale può essere un pattern prodotto da pochi agenti oppure un comportamento che emerge da pochi elementi del sistema, o uno o più eventi generati o scatenati dall'interazione di pochi agenti vicini. Due agenti sono

vicini secondo una dimensione spaziale, ovvero localmente vicini rispetto allo spazio in cui si sviluppa il sistema.

Globale è un aggettivo che assoceremo all'insieme di tutti gli agenti, o al sistema totale. Globale può essere un pattern generato da tutti gli agenti o una intelligenza emergente dal sistema in toto o l'insieme di tutti i processi che avvengono nel sistema. La globalità ha effetto sia nel luogo in cui vive il CS che sul suo ciclo di vita. L'effetto di un'interazione o un cambiamento locale ha quasi sempre conseguenze che si possono ripercuotere su tutto il resto del sistema o solo su un gruppo di altri agenti, dovuto dall'interdipendenza degli stessi.

Continuamente avvengono processi locali e globali in un CS. L'insieme di processi locali spesso genera uno o più effetti o processi globali. Si dice che un CS sia caratterizzato da una *causalità verso l'alto*, ovvero relazioni di causa-effetto locali che si ripercuotono in modo tipicamente non lineare nel sistema. All'opposto accade anche che fenomeni ed eventi globali si ripercuotono su proprietà locali del sistema, manifestando *causalità verso il basso*. L'influenza continua di eventi o processi locali verso quelli globali e viceversa contribuisce alla complessità dei sistemi.

Emergence

Intraducibile di per sé. "*Intelligenza emergente*" o "*comportamento emergente*" è l'espressione italiana che più si avvicina al concetto nato in terra statunitense. Per Emergence si intende la capacità di un CS di mostrare un insieme di comportamenti globali, non prevedibili o calcolabili semplicemente studiando i componenti del sistema. Emergence è l'affioramento di uno o più fenomeni dal sistema, non calcolabili a priori dalla sola osservazione degli elementi del sistema. Tale fenomeno complesso si manifesta a partire dalla più semplice interazione locale tra agenti.

Ogni CS prima elencato ha una forma propria di Emergence, chi producendo pattern inaspettati, chi apprendendo in modo decentralizzato e ogni volta originale, chi reagendo in funzione di input esterni ed interni, chi attuando meccanismi di difesa, chi evolvendosi nel tempo grazie a conoscenza appresa o evoluzione dell'ambiente di sviluppo. Un CS manifesta sempre un qualche tipo di Emergence nel corso della sua vita, è nella sua natura. Nel caso specifico di CAS, ci si riferisce ad Emergence come la capacità di far emergere dall'interazione di agenti una intelligenza globale non programmata, non centrale ma diffusa e condivisa, di cui ogni agente è direttamente e indirettamente responsabile. Si usa riferirsi ad Emergence come "comportamento" emergente verso CS non adattivi; mentre viene più usata un'accezione alla "intelligenza" emergente se ci si riferisce a CAS.

Non esiste un'unica modalità di manifestare comportamento od intelligenza emergente. Ogni CS è unico nel suo genere e mostra una qualche forma unica, particolare al sistema, di Emergence. Esso è tutto ciò che è possibile che un sistema

manifesti nella globalità a partire dalla più semplice interconnessione locale di agenti costituenti. Nei CS vale infatti la seguente regola aurea: *La somma delle parti non fa il tutto*.

Studiare le singole componenti del sistema secondo il classico approccio di semplificazione o riduzione non funziona, non fornisce nessuna comprensione su quale sia poi la natura del sistema globale e su che tipo di comportamento emerga da esso. Quel *"molto di più"* che manca alla sola somma degli agenti del sistema è proprio la sua Emergence. Una migliore equazione, espressa in linguaggio naturale, di come è composto ogni CS sarebbe:

$$\sum(agente + sue\ interazioni\ locali) = Sistema\ globale + Emergence.$$

Pattern

Spesso questa Emergence dovuta ad interazioni locali produce dei pattern riconoscibili ed espliciti o altri più velati o nascosti di cui non era possibile predire la loro manifestazione semplicemente studiando le singole interazioni. Pattern è un termine che anch'esso viene tradotto e declinato diversamente per ogni campo di studi. Qui definiamo pattern secondo le due solite dimensioni *spaziali* e *temporali*. Un pattern nello spazio è un'aggregazione di elementi che produce uno schema con una forma o caratteristiche definite; un pattern nel tempo sono degli eventi o comportamenti specifici e ripetuti.

Un pattern non è tale se si presenta una sola volta nel tempo o nello spazio ma solo se si *ripete* in posti diversi o tempi diversi. Un pattern può essere globale o semi-globale se si presenta come risultato rispettivamente dell'interazione di tutte le componenti del sistema o solo di alcune, quest'ultimo tipo di pattern è molto più frequente.

Precedentemente abbiamo elencato un gran numero di pattern prodotti da sistemi complessi. Si può notare ora come tutti loro hanno in comune la proprietà di essere forme, configurazioni o disposizioni ripetute nel luogo o nel tempo di vita di un CS.

Bottom-up behavior

Possiamo tradurlo con comportamento decentralizzato, distribuito e governato a partire dalle interazioni locali degli agenti, o emerso dalle fondamenta del sistema, dalle sue componenti alla base. È tra le caratteristiche primarie di ogni CS. Necessaria anche per far sì che un CS manifesti Emergence è che esso abbia una natura "*bottom-up*" (letteralmente dal basso verso l'alto), che l'emersione avvenga a partire dall'interazione dei suoi agenti semplici e non da regole centrali univoche, o strutture

globali prefissate. L'emergence di un sistema deve appunto emergere da esso a partire da regole locali agli agenti, e non venire programmata o prestabilita. È quindi il termine che racchiude la sintesi della struttura di un CS e indica dove risiede l'origine della sua complessità: a partire dal basso dai suoi comportamenti bottom-up.

Il comportamento bottom-up del sistema è ciò che crea meccanismi di causalità verso l'alto. Prima abbiamo detto che nei CS esistono anche meccanismi di causalità verso il basso. Il punto è che il comportamento bottom-up è propriamente caratteristico dei sistemi complessi, è uno dei suoi punti chiave.

L'opposto di bottom-up è un sistema top-down, dove tipicamente regna una struttura gerarchica con potere decisionale e di controllo sul sistema, oppure dove regnano delle regole imposte dall'alto su come deve essere il sistema e cosa deve fare. I sistemi top-down sono molto comuni nell'ingegneria, sono tali da permettere una formulazione unica del sistema, governata da regole centrali che ne stabiliscono il comportamento. Sono assolutamente essenziali anche in molte altre discipline come approccio a problemi deterministici e quantificabili, dei quali si vogliono dimostrare delle proprietà o garantire la presenza di certe qualità o stabilire un governo centrale del sistema.

Molti sistemi complessi presentano caratteristiche di entrambi. Ad esempio, la *politica* ha regole e leggi ben definite ed è costituita da un parlamento che le discute e le cambia e il loro effetto si ripercuote su tutta la popolazione, caratteristica di un sistema top-down. Eppure, la politica è un CASS, infatti, in essa si manifestano tipici comportamenti emergenti e "regole" auto prodotte e condivise da tutti gli agenti del sistema, senza che nessuno di essi le abbia decise o imposte direttamente, ad esempio come fare propaganda o come diffondere una idea e che effetto essa abbia. Un altro esempio chiave sono le *cellule* dell'essere umano. Esse hanno un DNA al loro interno che "detta legge" su cosa deve essere una specifica cellula, cosa fare e quando. Eppure, le cellule formano uno dei più complessi CAS in natura. Nessuna ha la visione dall'alto del funzionamento del corpo umano e nessuna governa tutte le altre. Alla nascita l'aggregazione di cellule produce pattern visibili - un braccio o un cervello - senza che nessuna di esse abbia deciso cosa fanno le altre, non c'è nessun piano superiore. Sono le interazioni tra cellule vicine a produrre tali pattern e comportamenti emergenti che danno forma e vita. Sono le interazioni tra cellule vicine a produrre un accordo e un adattamento condiviso, che genera un certo tipo di intelligenza emergente dal sistema.

Questo non vuol dire però che un CS non possa avere una struttura gerarchica al suo interno o non possa avere regole fisse, tutt'altro, i CS hanno bisogno di regole, di controllo e talvolta di gerarchie. La differenza è che non bastano solo quelle stabilite a priori ma anzi le più importanti e interessanti sono quelle emerse, distribuite e decentralizzate.

Sistema multi agente

Un sistema si definisce *multi-agente* (Multi-Agent System - MAS) semplicemente se è composto da una moltitudine di sottosistemi o agenti. Non tutti i MAS sono necessariamente dei CS. Troviamo MAS nel settore videoludico e cinematografico, ad esempio, che non necessariamente hanno le caratteristiche di un CS, come quelle che stiamo trattando in questo elenco. La CG (computer graphics) e la AI (Artificial Intelligence) fanno uso estensivo di MAS senza che essi sviluppino necessariamente caratteristiche di complessità.

Tutti i CS sono dei MAS, è proprio nella moltitudine che nasce la complessità, anche se essa è una caratteristica necessaria ma non sufficiente a garantire complessità. Esistono CS composti da poco più di un agente e questo basta a creare complessità, ad esempio in un sistema composto da tre pianeti che interagiscono per forza di gravità. Nell'insieme dei CAS è invece molto raro (forse non ne esistono) trovare sistemi adattivi non composti da un elevato numero di agenti. La componente adattiva del sistema richiede tipicamente un vasto numero di agenti per emergere. Rimaniamo vaghi sul numero specifico di agenti in un CS, continuando ad usare parole non formali come pochi e molti, dato che ogni sistema è a sé e non avrebbe senso stabilire numeri fissi validi per gruppi di sistemi diversi tra loro.

Obiettivi e strategia

Analizziamo ora un aspetto cruciale della presenza di CS in natura. La loro esistenza è profondamente collegata ad una naturale tendenza a raggiungere un *obiettivo*. Avere uno scopo o un fine è essenziale, dà significato a tutto. Gli obiettivi del sistema conferiscono una direzione ed un verso al sistema globale e questo tenderà a volerli raggiungere e compiere il più possibile. Senza obiettivi un CS cambierà difficilmente il proprio stato, non avendo un motivo intrinseco per farlo, perdendo probabilmente la sua natura dinamica e quindi rompendo la sua complessità.

Affinché il sistema rimanga complesso, l'esistenza di almeno un obiettivo è necessaria al suo sostentamento. Tuttavia, questo non è tutto, serve ancora un'altra componente al sistema: l'obiettivo stesso deve essere *dinamico*. Senza questo aggettivo associato ad almeno uno degli obiettivi del sistema, una volta raggiunto l'obiettivo, il sistema potrebbe degradare a sistema non dinamico, quindi non complesso.

Gli obiettivi possono essere preesistenti, costituiti alla nascita, oppure adottati nel corso di vita, talvolta anche auto prodotti. Per CAS l'esempio più naturale è: rimanere in vita; operare quindi tutte le azioni necessarie alla sopravvivenza del sistema, prime fra tutte nutrirsi e riprodursi. L'obiettivo di rimanere in vita è altamente dinamico, cambia continuamente, non serve solo a far sì che il sistema sia vivo nell'attimo presente, ma comporta mettere in atto una serie di funzioni affinché il sistema punti

alla sua sopravvivenza futura, preparandosi ad ogni possibile stravolgimento dello stato presente.

Questo si collega al concetto di *strategia*: ogni CS ne definisce almeno una per il conseguimento dei suoi obiettivi. Ovviamente tale strategia è spesso frutto di un comportamento bottom-up, nessuno nel sistema la decide e la impone agli altri, è una scelta che emerge dalle necessità condivise e distribuite nel sistema.

Storia e Memoria

La complessità di molti CS è fondata sulla loro *storia*, in base alla quale orientano le proprie scelte, ovvero di eventi accaduti nel passato, di informazioni memorizzate o esperienze fatte, magari per raggiungere più facilmente propri obiettivi, migliorare il proprio stato o anticipare eventi futuri. Non conosciamo ancora perché questo avvenga e come, ma sappiamo che la storia gioca un ruolo importante nella vita di un CS.

Alla storia è profondamento collegato un altro concetto chiave. Essa necessita di un meccanismo di *memorizzazione* e consultazione. Tale memorizzazione avviene in modo quasi sempre decentralizzato. Ogni agente non è mai totalmente cosciente di tutti i processi che avvengono nel sistema e tanto meno della sua memoria globale. Essa è costruita sulla base dell'insieme condiviso di valori e processi ricordati da ogni singolo agente, l'unione della memoria di ogni agente crea la memoria totale del sistema.

Faremo tuttavia riferimento alla memoria del sistema e all'affidamento alla sua storia come se il sistema possedesse una risorsa centrale unica e ben definita. Questa semplificazione risulta efficace ogni volta che ci si riferisce a comportamenti globali.

Interdipendenza, influenzabilità e imprevedibilità

Si è parlato già di interdipendenza, influenzabilità ed imprevedibilità nel contesto dei sistemi dinamici non lineari. Qual è il significato che attribuiamo a queste caratteristiche rispetto ai CS?

Analizziamo queste tre parole, a loro volta connesse, nei due soliti punti di vista: locale e poi globale. Per *interdipendenza* e *influenzabilità* locale in un CS si intende la dipendenza *reciproca* tra agenti vicini nel sistema. Lo stato di un agente dipende ed influenza lo stato degli agenti vicini in funzione delle comunicazioni e dei feedback scambiati vicendevolmente. Ogni agente ha normalmente vicini diversi, un cambiamento di stato in un agente può influenzare un cambiamento in un altro suo vicino e questo ad uno dei suoi, come si può immaginare, crea una *catena* di dipendenze che si ripercuote globalmente nel sistema, generando forme, delle più

disparate di comportamento emergente. In un sistema altamente connesso ogni agente è quindi direttamente dipendente e influenzabile dai suoi vicini e indirettamente dipendente da ogni altro agente del sistema.

Questa rete *diretta* e *indiretta* di influenzabilità e interdipendenza causa una *imprevedibilità* globale nel sistema. È imprevedibile una risposta ad un nuovo input capace di causare una serie di effetti a cascata nel sistema. Addirittura, è possibile che il sistema si comporti in modo del tutto differente in due occasioni distinte in reazione agli stessi input, se la propagazione degli stessi avviene anche solo in modo leggermente diverso nelle due volte. La proprietà di imprevedibilità è spesso associata a sistemi caotici, tuttavia è anche presente in quasi tutti i sistemi complessi non caotici.

Nella cultura moderna ci si riferisce a questo insieme di fenomeni come *"Effetto Farfalla",* tipico dei sistemi caotici.

Su questo tema sono stati girati film, scritti libri e articoli scientifici riferendosi ad una nota provocazione lanciata dal matematico Edward Lorentz, il quale, idealmente, ipotizza la possibilità che "*un battito d'ali di una farfalla in Brasile possa scatenare un uragano in Texas"*. L'affermazione è chiaramente un'esagerazione dei comportamenti dipendenti di un CS, ma *non* è falsa. Non è infatti proprio possibile quantificare gli effetti di un minimo evento sul comportamento globale di un CS.

Molti effetti globali devono la loro manifestazione alla natura fortemente interconnessa, interdipendente ed influenzabile del sistema. I pattern che si generano dipendono da questa natura dei CS, la sua stessa Emergence ne è direttamente connessa. La complessità risiede quindi anche in queste tre parole chiave e da esse dipendono molte altre proprietà che stiamo elencando e analizzando.

Adattamento

Conseguenza e necessità di forti cambiamenti a cascata nel sistema determinano la dr capacità di adattarsi ad essi. L'*adattamento* è una parola chiave, tanto da essere una proprietà di confine tra CS e CAS (è proprio la "a" in CAS).

Abbiamo prima distinto CS da CAS in base al concetto di adattabilità a cui abbiamo dato una prima definizione. Ricordiamo che un CAS è adattivo se sa reagire agli stimoli che riceve, ai cambiamenti e alle modifiche del suo stato iniziale operando principalmente due azioni chiave in funzione di esse: *imparare* ed *evolvere*.

Tali modifiche al sistema possono essere temporali: un evento che accade, un nuovo stimolo esterno, un cambiamento interno; o spaziali: uno spostamento di uno o qualche agente del sistema, uno spostamento delle risorse o degli obiettivi.

Ci possono essere varie forme e modalità di adattarsi ai cambiamenti come ad esempio: un adattamento all'ambiente mutevole di vita, un adattamento

all'invecchiamento del sistema, un adattamento alla morte o alla nascita di nuovi agenti del sistema, un adattamento ad un cambiamento di stato del sistema, o ai mezzi comunicativi degli agenti.

Come e perché un CAS si adatta? Un CAS si adatta modificando il suo stato ed il suo comportamento globale e locale. Localmente un singolo agente può cambiare il suo stato o il modo in cui si connette con altri agenti per adattamento, globalmente operando modifiche su alcuni o tutti gli agenti o gruppi di essi o mostrando pattern diversi. Possono esistere quindi modalità di adattamento multi-livello in funzione del livello gerarchico interno al sistema. Generalmente ci sono sempre almeno due livelli: locale e globale. Ovvero esiste un adattamento dei singoli agenti in base alla loro natura, ai loro stati, le loro connessioni, in conseguenza a dei cambiamenti, reazioni ad eventi locali; ed esiste un adattamento globale del sistema, emerso dall'insieme di processi adattivi prodotti dai suoi agenti. Non dobbiamo pensare questa struttura multi-livello a compartimenti stagni, all'opposto ogni livello influenza gli altri. L'adattamento bottom-up al livello più basso influenza l'adattamento globale al livello più alto e viceversa. La causalità verso il basso e verso l'alto ricordiamo essere sempre presenti in CS e influenzare molti, se non tutti i suoi processi, tra i quali, appunto, l'adattamento in CAS.

Il motivo dell'adattamento è invece più complesso. I CAS si adattano prima di tutto perché sono sistemi dinamici quasi sempre vivi. Necessitano, quindi, di un continuo adattamento alle condizioni perennemente mutevoli della loro stessa struttura e dell'ambiente in cui sono immersi. Sebbene ciò sia vero, l'adattamento non è semplicemente una risposta dinamica e continua a stimoli e variazioni di stato ma è legata a motivi più intrinsechi. Primo e più importante fra essi è il bisogno del sistema di tendere continuamente ad uno *stato globale ottimale*. Se non si adattasse un CAS rimarrebbe sempre nello stesso stato senza mai raggiungere una condizione a lui maggiormente favorevole. È qui che si gioca tutto. Tendere a migliorare sempre il proprio stato in funzione di un ambiente dinamico e di sconvolgimenti continui che necessitano quindi, una ricerca *senza fine* di un possibile stato ottimale o semplicemente migliore.

L'insieme dei processi messi in atto per adattarsi è spesso infinito, uno stato ottimo globale raramente viene raggiunto, sia che esso non esista o che non possa essere raggiunto, ma tratteremo questo più avanti quando si parlerà di non ottimalità.

I due processi chiave (apprendimento ed evoluzione, con la possibile presenza di un terzo elemento) costituenti dell'adattamento di un CAS, sono così importanti da meritare una trattazione specifica.

Apprendimento

Per adattarsi in maniera ottimale all'ambiente ed alle situazioni che li circondano, i CAS sviluppano meccanismi di *apprendimento*, inteso come l'insieme delle funzioni di raccolta delle informazioni sul proprio stato iniziale e su eventi presenti e passati che vengono elaborati per ricordarle in futuro ed evolvere in funzione di esse. Abbiamo parlato prima di storia e di memoria nei CS. Nell'ambito dei processi di apprendimento, l'affidamento alla memoria ed alla propria storia è una componente fondamentale, esse permettono la classificazione e il riconoscimento di pattern già visti e formano il materiale di base per studiare una strategia per il futuro che tenda a migliorare il processo di adattamento. Per adattarsi quindi i CAS hanno bisogno di sapere ciò che esiste nel loro ambiente o conoscere gli eventi che accadono e reagire ad essi, imparare svolge quindi la funzione di migliorare il comportamento futuro del sistema.

Il problema, tuttavia, risiede nel fatto che sappiamo cosa si intende per apprendimento ma abbiamo pochi strumenti per misurarlo in un CAS. Ovvero è difficile capire quando e in che quantità un CAS riesce ad imparare da ciò che accade intorno ad esso e adattarsi meglio all' ambiente di sviluppo. Questo è un processo chiave, e tuttora oggetto di notevole ricerca scientifica.

Evoluzione

Seconda componente chiave della definizione di adattamento di un CAS. Per *evoluzione* si intende la proprietà di un CS adattivo di mutare nel tempo e nello spazio cambiando: forma, configurazioni, stati, processi, interconnessioni, agenti; ovvero possibilmente tutto ciò di cui è composto. L'evoluzione è la conseguenza naturale di un CAS vivo, si può considerare *morto*, infatti, un CAS che rimane statico e uguale per un periodo di tempo indefinito.

Occorre fare però una distinzione importante tra cambiamento ed evoluzione. L'evoluzione, nella Teoria dei Sistemi, è intesa come un cambiamento per uno scopo o verso un obiettivo. Ogni cambiamento dunque, non è fine a sé stesso. Nonostante il CAS decida o no, sia cosciente o no, di avere o scegliere obiettivi essi sono presenti e costituiscono l'impulso grazie al quale tutto il processo evolutivo ha inizio. Questo perché essi tendono sempre verso stati e condizioni migliori di vita, a metodi di adattamento preferenziali in funzione dello stato presente e alle condizioni ambientali.

Apprendimento ed evoluzione sono intimamente collegati e collaborano nel processo adattivo. L'evoluzione è infatti conseguenza diretta di un apprendimento. Questo processo di apprendimento ed evoluzione che chiamiamo adattamento è costante e continuo. Non si arresta mai se non con la morte del CAS.

Anticipazione

Esiste una possibile terza componente dell'adattabilità di un CAS, essa è l'*anticipazione*. Se i meccanismi di apprendimento sono ancora per lo più oscuri alla teoria della complessità ancor di più lo è la capacità di un CAS di anticipare eventi o situazioni future. Intendiamo come anticipazione, infatti, la capacità di un CAS di prevedere sviluppi futuri di sé stesso, dell'ambiente circostante o di processi ed eventi che potrebbero accadere, predisponendo in anticipo, una serie di funzioni atte a gestire tali previsioni. È una proprietà tipica dell'adattamento anche se è ancora poco chiaro come ciò avvenga.

Esistono tuttavia ricerche nelle quali sono stati osservati sperimentalmente gli effetti di anticipazione in CAS reali. Ad esempio, la capacità di un formicaio di prevedere nella stagione estiva, il fabbisogno di cibo per l'inverno, quando le necessità saranno totalmente diverse dal presente. Ovviamente nessuna formica da sola sa di anticipare i processi che avverranno d'inverno, ma è la collettività, quindi il formicaio, che è un CASS, a decidere, principalmente per regole bottom-up, di attuare tutte quelle attività che porteranno il formicaio ad uno stato ottimale per affrontare un futuro bisogno. L'anticipazione è fortemente collegata ai processi di apprendimento e di evoluzione di un sistema complesso; quindi, collegata al macro-concetto di adattabilità di un CAS, sebbene sia una delle proprietà più complesse e misteriose dei CAS.

Non ottimalità

Discutiamo brevemente cosa intendiamo allora per *stato* di un CAS e per *stato ottimale*. Uno stato consiste in un'istantanea di un momento specifico nel tempo delle condizioni di un CAS. È l'insieme di informazioni, che si possono raccogliere e codificare, in un istante preciso di un CAS. Ad esempio, quelle informazioni che codificano uno stato possono essere il numero di agenti comunicanti del sistema, i loro collegamenti e la loro configurazione o i processi in atto in un determinato momento. Lo stato di un sistema è quindi un concetto astratto, il cui valore dipende dall'osservatore, non è una proprietà intrinseca. Uno stato ottimale è quindi quello stato che ha l'insieme di valori perfetti secondo una scala di valori stabilita dall'osservatore.

Generalmente un CAS, non dovrebbe mai raggiungere uno stato ottimale perché questo vorrebbe dire non tendere più ad evolvere per raggiungere stati migliori: se non evolve non si adatta, quindi perde di interesse.

Può però accadere che un CAS, arrivi a stabilizzarsi su uno stato ottimo. È un caso raro ma non impossibile e non deleterio per il sistema. Molti sistemi complessi prevedono dei punti di equilibrio che possono essere talvolta codificati come stati ottimali per essi. Tuttavia i sistemi altamente dinamici tipicamente vivono in un ambiente esso stesso dinamico ed in continua evoluzione. Questo implica che è ancora più raro che

uno stato ottimo non cambi nel tempo, provocando nel sistema la ricerca di una diversa strategia adattiva per raggiungere il nuovo stato ottimo. Qualora il CAS diventasse un sistema statico, non adattandosi più a nuovi cambiamenti di stato interni o del suo ambiente, significherebbe perdere le sue proprietà di adattabilità oppure morire, comunque sia non sarebbe più un CAS.

Vedremo nei prossimi capitoli come l'approccio alla modellazione più adatto per i CAS non consista nel trovare delle soluzioni ai suoi modelli, dato che potrebbero anche non esistere, o nel trovarne un ottimo locale (secondo una terminologia tipica dell'Intelligenza Artificiale); ma nello studiarne il "*process of becoming*" (John Holland), ovvero il processo di divenire, il percorso di vita del sistema. In questa disciplina il viaggio è ben più importante della meta. Il focus non è quindi negli obiettivi del CAS ma in tutto il suo processo evolutivo, nella sequenza di stati attraversati, nella sua storia e nel suo futuro, sperando che tutto questo non si arresti mai in un ottimo locale o globale.

Riproduzione

Nessun agente ha una visione globale sul sistema, ognuno ha solo una porzione di conoscenza e di informazioni, ma è essenziale per partecipare ad uno degli effetti globali del sistema. Queste informazioni necessitano di muoversi in un ambiente dinamico, e ciò avviene principalmente in due modalità: per comunicazione e per riproduzione. Ci occuperemo di comunicazione al prossimo paragrafo. Entrambe le modalità avvengono in modo continuo contribuendo ad arricchire il bagaglio di informazioni che il sistema ha bisogno per vivere e per adattarsi al meglio.

Ogni essere vivente può essere accomunato da un istinto di sopravvivenza e dalla capacità di *riprodursi*. Per un CAS composto da agenti vivi, il riprodursi è più di un bisogno, è una fase di vita essenziale. La riproduzione è la conseguenza principale del suo processo evolutivo.

Questa può avvenire in varie forme e modalità, come ci insegna la biologia, ma non ci occuperemo di questo, quanto dei suoi effetti, come si ripercuote nel sistema, che emergence avviene.

Tra gli aspetti fondamentali e più studiati della riproduzione vi è la capacità di trasmettere *conoscenza* dai genitori ai figli. Qui intendiamo genitore qualunque agente del sistema che partecipa alla nascita di un nuovo agente nel sistema. Il passaggio di conoscenze tra generazioni è ciò che permette al sistema di operare il suo infinito processo di adattamento nel percorso di raggiungimento degli obiettivi.

Affascinante è osservare un CAS nel susseguirsi, totale o parziale, dei suoi agenti per riproduzione (nuove nascite e morti), rimanendo fondamentalmente lo stesso, con gli stessi obiettivi e processi. Il paradosso della *nave di Teseo* esprime proprio il dubbio

metafisico sul problema di un'entità composta da più elementi: cambiando ogni sua componente costitutiva nel tempo essa rimane fondamentalmente la stessa. Basti pensare al corpo umano fatto di miliardi di cellule che continuamente nascono e muoiono eppure, nel breve tempo, noi rimaniamo gli stessi senza notare alcuna differenza. Questo continuo nascere e morire, nell'arco della vita di un essere umano è ciò che stabilisce le fasi di vita del CAS: mantenere gli obiettivi sempre aggiornati, regolare il passaggio di stati, spingere il sistema verso stati sempre migliori, in risposta ad un ambiente sempre in cambiamento.

Comunicazione

Un CAS è caratterizzato da una moltitudine di agenti interconnessi. L'*interconnessione* non è solo logica, o fisica, o temporale, è molto di più. L'interconnessione stessa è all'origine della sua complessità, è ciò che genera comportamenti emergenti. Ma cosa c'è alla base di questa rete di connessioni? La *comunicazione*. Gli agenti del sistema comunicano continuamente tra loro e questo è veramente il carburante che scorre nella rete di connessioni e quindi ciò che causa la complessità e l'emergence.

La comunicazione tra agenti del sistema avviene in modi diversi per ogni CAS. In generale si tratta di uno scambio di informazioni tra agenti vicini. Ribadiamo il concetto di vicinanza sempre inteso nelle 4 dimensioni in cui siamo abituati a vivere, ovvero le 3 spaziali e quella temporale; quindi, due agenti sono vicini nello spazio o nel tempo o in entrambi. Tipicamente nessun agente comunica con tutti gli altri agenti del sistema, lo fa solo con i suoi vicini. Tuttavia, la comunicazione locale tra agenti vicini, ancora una volta, ha effetti globali. Questo perché ogni comunicazione avviene in un ambiente dinamico, non lineare, animato da agenti interdipendenti. Una informazione passa facilmente da un agente ai suoi vicini; questi ritrasmetteranno tale informazione, magari arricchita dalle proprie conoscenze, ai loro diversi vicini, e così via a cascata. Una informazione può attraversare tutto il sistema passando da un agente all'altro, ripercuotendosi globalmente.

Ancora una volta le leggi di comunicazione sono bottom-up. Nessuno ha stabilito la forma, la modalità o i tempi in cui essa debba avvenire. Ogni agente ha le sue piccole regole di comunicazione. Eppure, vivendo in un sistema complesso questo provoca effetti non lineari nel sistema. È la non linearità delle comunicazioni a provocare effetti di intelligenza emergente in CAS, a trasmettere conoscenza all'interno del sistema, a stabilire le regole di evoluzione e scegliere come passare dallo stato presente a quello successivo, possibilmente migliore. Queste scelte avvengono in modo globale nel sistema attraverso questa complessa rete di interconnessione di agenti comunicanti, e le informazioni che si passano sono di vari tipi.

Ogni comunicazione è quindi essenziale al sistema, perdersi anche una sola informazione può avere effetti a cascata esponenziali ed imprevedibili dato che ogni

piccolo pezzo di informazione trasmesso confluisce nell'insieme delle informazioni che attraversano il sistema collaborando a creare intelligenza emergente.

Feedback

Ci manca descrivere un aspetto della comunicazione degli agenti interconnessi: i *feedback*. Ogni agente non comunica solo per dare informazioni ma anche per riceverne. Questo è il feedback, l'effetto risultante dalla comunicazione ricevuta che si riflette sul sistema stesso per variare o correggerne opportunamente il funzionamento. Accade spesso il formarsi di pattern di comunicazioni cicliche definiti "*feedback loops*", cioè quando una informazione viene ritrasmessa all'agente che l'ha generata dopo aver subito una rielaborazione da parte degli agenti vicini, giungendo in una forma "evoluta" rispetto all'originale, questi loop possono essere di due tipi: positivi o negativi.

Un "*positive feedback*" o "*feedback rinforzante*" è un loop di feedback in cui un messaggio o un'informazione amplifica il messaggio e l'informazione originale, secondo certi valori o parametri. È un feedback positivo quando la risposta segue l'andamento di amplificazione o diminuzione aumentando o diminuendo rispettivamente i valori o gli effetti del messaggio di feedback. Positivo è proprio perché mantiene il segno dell'operazione precedente sommando nuovi effetti. Dati due agenti A e B, A comunica a B un messaggio, B lo amplifica in qualche modo e lo ritrasmette ad A che a sua volta, lo amplifica ulteriormente per poi re-inoltrarlo a B. Un esempio di questo tipo è costituito dalle oscillazioni di un ponte in seguito a traffico o terremoti, se tali oscillazioni entrano in fase, in un loop di feedback positivi, possono portare ad effetti collaterali distruttivi. I feedback positivi contribuiscono quindi all'instabilità del sistema secondo legami di causa-effetto aventi risultati esponenziali nel sistema, possono portare il sistema in stati caotici e progressivamente più lontani da stati di equilibrio. Sono fortemente presenti nei sistemi caotici.

Un *"negative feedback*" o "*feedback smorzante*" è il suo opposto, ovvero la diminuzione degli effetti di un messaggio ricevuto. Dati due agenti A e B, A invia un messaggio a B che lo riceve, lo interpreta, lo modifica e lo rinvia ad A, il quale in base al contenuto del messaggio di feedback se esso è aumentato, secondo un certo parametro, lo diminuirà in intensità o in valori, se invece è diminuito lo amplifica. È negativo nel senso che cambia il segno del messaggio di feedback rispetto ai suoi valori o effetti. Un esempio di questo tipo è la regolazione dello zucchero nel sangue mediante la secrezione di insulina che ne regola le quantità, oppure il funzionamento di condizionatori che, in funzione della temperatura impostata, immettono aria fredda o calda a seconda della differenza con la temperatura ambientale. I feedback negativi portano quindi il sistema verso uno stato di equilibrio e di stabilità, o minimizzano la differenza tra lo stato di un sistema ed uno stato obiettivo.

L'*equilibrio* dei due tipi di loop è necessario alla sopravvivenza di qualsiasi CS. La presenza di un solo tipo di feedback loop infatti è indicativa di una perdita di proprietà intrinseche dei sistemi complessi. Un tipo serve a mantenerne la complessità o la sua natura caotica e contribuisce all'instabilità ed alla imprevedibilità del sistema, portandolo a non raggiungere mai stati ottimali. L'altro serve a contrastare il caos e mitigarlo, serve a spingere il sistema verso un continuo miglioramento, verso il processo di stabilizzazione e scelta delle minime risorse e processi necessari ad evolvere il sistema dallo stato presente ad uno stato futuro migliore, contribuendo alla stabilità del sistema al fine di mantenere un minimo grado di equilibrio interno.

Cooperazione vs Competizione

Nell'ambito di un CS, un agente può assumere due fondamentali stati distinti nel tempo o nello spazio in relazione agli altri agenti del sistema: lo stato *cooperativo* o lo stato *competitivo*. Si assume uno dei due comportamenti in base alle regole interne dell'agente, in funzione di una comunicazione o di un feedback, oppure in risposta ad un nuovo input o un cambiamento dell'ambiente, oppure per adattarsi al meglio.

Spesso vengono a formarsi nel sistema pattern formati da gruppi di agenti legati dallo stesso comportamento cooperativo o competitivo. Questo può avvenire ad esempio perché è emersa una necessità di aggregazione per portare il sistema, ad esempio, verso uno stato migliore. Inoltre, per il benessere del sistema, può anche capitare che in un gruppo siano presenti agenti con comportamenti opposti. Infine, sono da considerare fondamentali anche le unioni e le comunicazioni tra gruppi di agenti in stati uguali, opposti o misti, salendo quindi a livello globale. Come sempre le unioni e le scelte di comportamenti adottati, da un singolo agente o da un gruppo di essi al fine di migliorare la qualità della vita del sistema, sono stabilite da regole bottom-up.

Può venire naturale chiedersi perché sia necessaria la presenza di entrambi gli stati in uno stesso sistema, anziché avere tutti gli agenti in stato cooperativo. La risposta non è affatto ovvia, dato che la scelta di uno stato è dettata da regole bottom-up; una visione del sistema dall'alto non ci aiuta a capire quale sia la necessità di avere agenti in competizione fra loro. Eppure ciò avviene, tipicamente perché al sistema conviene, magari per raggiungere uno stato migliore. Nei CAS, è affascinante osservare come gli agenti non scelgano uno stato per il bene del sistema globale, ma solo perché localmente conviene così. La distribuzione e l'unione e il numero di caratteri diversi sono caratteristiche emergenti, dettate da necessità implicite nel sistema. Si pensi all'equilibrio tra prede e predatori in un ecosistema, il numero di entrambi si regge su un equilibrio, il cui scompenso provocherebbe gravi e talvolta irreversibili sconvolgimenti al sistema. È quello che sta accadendo in molti ecosistemi nei quali l'influenza dell'uomo ha portato a una diminuzione drastica di tipologie di agenti causando cambiamenti radicali e dannosi.

Occorre precisare che entrambi gli atteggiamenti, cooperativo e competitivo, possono causare sia un feedback positivo che negativo, a seconda dello scopo da raggiungere. Ad esempio, un feedback positivo può essere prodotto da gruppi di agenti in cooperazione per aumentare progressivamente una risorsa, o un processo. Feedback negativi possono essere prodotti da agenti in competizione per produrre effetti oppositivi tra loro. Oppure un feedback loop negativo può essere prodotto da agenti in cooperazione per mantenere uno stato, una risorsa o un processo in equilibrio. Infine, per competizione, si possono creare feedback loop positivi per andare a peggiorare od amplificare un processo, causando instabilità per l'agente oppositore o per indurlo a adottare un comportamento desiderato.

I motivi di una singola scelta locale o globale sono talvolta oscuri all'osservatore, questo è tipico dei CS. In senso figurato bisogna "*allontanarsi*" dalla figura e osservarla nel suo insieme per capire, esattamente come di fronte ad un quadro impressionista (Emergence - John Holland).

Aggregazione e Ricombinazione

Abbiamo accennato prima al fenomeno di aggregazione tra agenti, vediamo cosa intendiamo nel dettaglio. Un agente, singolarmente, può avere un minimo effetto sul sistema, ha bisogno di altri agenti per una serie di funzioni come comunicare per apprendere, ricevere feedback dei suoi messaggi, adattarsi ad un ambiente circostante sempre in cambiamento.

Questi "*cluster*" di agenti sono necessari per suddividere un CAS in sottosistemi complessi più semplici, focalizzati su minori obiettivi e più specifici nello svolgere funzioni chiave del sistema. Si prendano per esempio i tessuti del corpo umano: ognuno di essi è un CAS, sottoinsieme del CAS più grande che è il corpo umano. Ognuno svolge funzioni specifiche ed ha obiettivi di minore entità di quelli del sovra sistema.

Ma semplificazione e specializzazione non sono l'unico motivo. Aggregati di agenti si formano anche per compatibilità di regole, di atteggiamenti o comportamenti affini, per tipologia di messaggi mandati, per età o per una varietà di altri motivi, allo scopo di migliorare la condizione dell'agente o di un gruppo di agenti, l'aggregazione vale infatti anche tra gruppi, non solo tra singoli.

Tuttavia nulla è immutabile in un CAS, ogni secondo avvengono eventi che portano a piccole, continue modifiche del sistema. Questo causa la *ricombinazione*. Ovvero la separazione di un agente o di un gruppo da un cluster per ricombinarsi in un altro modo o con un altro cluster, è sintomo della vivacità di un CAS. Il continuo aggregarsi, sciogliersi e ricombinarsi in modo diverso, viene spesso visto come se il sistema

"*respirasse*", e questo continuo respiro produce talvolta pattern semi globali o globali e peculiari comportamenti emergenti.

Self organization

Tutte le caratteristiche finora descritte si accomunano sotto il termine ombrello di "*self organization*", che traduciamo come auto organizzazione. Rappresenta la capacità di un CAS di organizzarsi in maniera autonoma, decentralizzata, auto condivisa, tendente all'ottimo e adattiva. È quindi una caratteristica che possiamo definire riassuntiva di tutte le proprietà dei CAS.

I ricercatori indicano questa come una delle chiavi per capire profondamente un CAS, qualcosa di strettamente legato al concetto di bottom-up behavior. Infatti, un CAS si auto organizza su un insieme di regole nate "dal basso" la cui globalità rende possibile l'auto organizzazione e l'adattamento dell'intero sistema.

Tutto ciò che accade in un CAS, risponde all'unica regola condivisa di auto organizzarsi, ma qui accade la magia: in realtà questa stessa regola si *auto produce*. Non è stata preimpostata nel sistema, o scelta da qualcuno o qualcosa e nemmeno dagli agenti, emerge dal sistema anche se è essa stessa la causa dell'emersione. Come direbbero i ricercatori: "*it just happens*" (succede e basta). Sembra un paradosso ma è la natura complessa propria del sistema a creare questo loop.

Pensiamo, ad esempio, alle risposte condivise nell'affrontare una emergenza nel sistema o nell'ambiente circostante, oppure per rispondere ad una necessità di evoluzione o di adattamento, o nello stabilire quando, come e dove reperire le risorse necessarie al sostentamento del sistema ed infine nello stabilire gli obiettivi e come raggiungerli. Tutto ciò il sistema è in grado di auto produrlo, si organizza in modo autonomo e decentralizzato per soddisfare uno qualsiasi dei suoi bisogni.

Ogni CAS si può dire che si caratterizzi proprio dal modo in cui esprime queste facoltà di auto organizzarsi, quali regole auto produce, quali proprietà fa emergere, che intelligenza emerge dal sistema e, in ultima analisi, da che pattern si possono distinguere nel sistema.

3. Modelli computazionali per CAS

Introduzione

Modellare un CAS è un'operazione ardua a causa della loro natura instabile e mutevole nel tempo e nello spazio. La loro tendenza evolutiva e di adattamento rende impossibile o impraticabile modellarli per via classiche o approcci tipici per tutti gli altri sistemi.

Il primo scoglio che si incontra è quindi creare un modello che simuli la complessità e l'adattività del sistema modellato. Il modello non dovrebbe avere una unica soluzione fissa nel tempo, non deve avere un unico punto di equilibrio ottimale in ogni situazione, non può esserci uno stato ottimale costante nel tempo. Se ci fosse, molto probabilmente il modello sarebbe troppo semplice, oppure non modellerebbe correttamente la realtà. Come fare ad evitare che il modello si areni su uno stato ottimo nel tempo o su un unico punto di equilibrio? Adotteremo a questo proposito diverse tecniche per garantire il rispetto di questa proprietà che garantisce un livello minimo di qualità.

Un'altra difficoltà risiede nella unicità sostanziale di ogni CAS. Seppure abbiano tutti caratteristiche e funzioni simili per natura, sono sistemi altamente variabili e mutevoli. Approcciarsi ad essi e tentare di modellarli deve quindi seguire un comportamento ad hoc per ognuno. Per risolvere questo problema non tenteremo di modellare un CAS qualsiasi ma adotteremo scelte ben precise sulla sua natura.

Un breve sommario su quale sia l'approccio consigliato dai ricercatori nello studio della Teoria della Complessità, ma ancor di più nello specifico di CAS, è di costruire modelli computazionali sulla base della teoria studiata. Modelli che siano quindi digeribili per un computer ed orientati alla computazione, preferibilmente altamente paralleli e interagibili, composti da un alto numero di agenti governati da semplici regole che hanno effetto locale sul sistema, e poi far evolvere il sistema autonomamente o come direbbero gli scienziati del settore: "just run it and see what happens" (J. Holland – S. Johnson – Miller and Page).

Finora abbiamo analizzato in maniera estensiva e dettagliata tutte le caratteristiche principali di CAS da un punto di vista prettamente scientifico, non necessariamente indirizzato alla modellazione, anche se è sempre stato il nostro scopo fin dall'inizio. Ci sono tuttavia altri aspetti dei CAS di cui non si è ancora trattato ma essendo delle peculiarità orientate propriamente alla loro modellazione si è ritenuto fosse più adatto trattarne solo ora. Naturalmente esse si appoggiano sulla base teorica fin qui costruita, la quale risulterà sempre fondamentale per qualsiasi riferimento e consulto durante la modellazione.

Per quanto la modellazione computazionale di CAS segua metodologie ad hoc e non standard è comunque necessario stabilire un insieme minimale di proprietà e qualità che il modello ha bisogno di avere per poter essere valido, ma ancor più per essere scientificamente interessante e fonte di studi.

Successivamente esponiamo le caratteristiche desiderabili in un modello di CAS. Vige il principio d'oro dell'*equilibrio*. Troppe regole e proprietà da rispettare limitano il modello a rappresentare solo quello che gli si chiede e nulla di più. Si impedisce l'esplorazione e la variabilità, che sono invece estremamente importanti per modelli rivolti a sistemi complessi. Viceversa, troppe poche regole e troppa libertà conferita al modello lo farebbero divergere enormemente dai casi di studio e dai risultati attesi, il caos e la casualità sarebbero talmente elevati dal produrre un nulla di fatto. La difficoltà è quindi trovare il giusto equilibrio, consci che la ricerca di tale punto è un processo lungo e richiede una grande quantità di prove ed errori per affinare sempre più il modello alla realtà, alla natura, ed agli studi raccolti.

Modellare la complessità

Se volessimo fare un sunto delle proprietà intrinseche di un CAS diremo che esso è caratterizzato da un insieme di elementi interconnessi, interdipendenti e comunicanti. È dinamico, imprevedibile e talvolta caotico. È governato da regole bottom-up e da relazioni di causa-effetto di tipo dal basso verso l'alto e viceversa. È in grado di auto organizzarsi, adattarsi, riprodursi, ricordare il proprio passato e attuare strategie per il conseguimento di obiettivi futuri. Soprattutto manifesta Emergence delle modalità più disparate, tra le quali l'intelligenza collettiva o l'affioramento di pattern.

La complessità non è data solamente dalla natura dei suoi agenti, come sappiamo, ma molto più dalle modalità delle loro interazioni reciproche.

Modellare un CAS è assai complesso a sua volta. È altamente improbabile tentare di simulare tutti gli aspetti che li caratterizzano, risulta quindi necessario selezionare alcune loro proprietà o astrarre alcuni processi specifici.

Purtroppo emergono seri problemi quando si tenta di semplificare un CAS. Ad esempio potremmo pensare di ridurlo e semplificarlo. Ma cosa accade ad un CS se tento di semplificarlo togliendo un agente dal sistema? Il modello rispecchia con fedeltà la complessità del sistema? Quanto posso semplificare il problema prima di romperlo e distanziarmi irrimediabilmente da esso? Un sistema complesso è caratterizzato da agenti interdipendenti e influenzanti. Rimuovere un elemento dal sistema non lo rende meno complesso, il sistema rimane comunque complesso ma può cambiare profondamente la sua struttura. Esistono CS molto fragili nei quali la separazione di un solo suo agente può compromettere tutto il sistema. La rete di dipendenze, quando privata di un suo elemento, provoca effetti a cascata non lineari in grado di rompere definitivamente il sistema. Esistono invece CS più robusti nei quali

la perdita di un suo agente non compromette la sua sopravvivenza o la sua integrità. Tuttavia, la separazione produce effetti che si propagano nel sistema da un elemento al vicino successivo. Il sistema può mutare profondamente la sua natura anche in risposta di un minimo evento. Più in generale si mettono in moto una serie di processi adattivi per compensare la perdita e per riorganizzare autonomamente il sistema, in modo che sia in grado di mantenere la sua natura il più possibile. In particolare varia da sistema a sistema la risposta a meccanismi di semplificazione, alcuni sono più sensibili di altri.

Risulta quindi difficile stabilire con certezza se un modello che semplifica il sistema ne cattura profondamente la sua complessità. Eppure qualche astrazione sarà necessario farla a meno di disporre di super computer per modellare nel dettaglio alcuni CAS, come stanno facendo alcune aziende.

Modellare la complessità è quindi un'operazione ardua che si occupa di trovare degli equilibri, dei compromessi attraverso molte prove ed errori e raffinamenti continui.

Regole

Un primo aspetto cruciale nella modellazione di CAS è quale debba essere la struttura delle regole che lo governano. L'obiettivo è quello di modellare un sistema governato da regole bottom-up. Quindi l'approccio corretto è quello di dare un minimo insieme di regole agli agenti del sistema e poche o nulle regole generali top-down sul suo comportamento globale.

L'ambiente esterno del sistema può essere progettato secondo un approccio top-down, semplificando le modalità di interazione del sistema con esso. Tuttavia, tutto ciò che concerne il sistema necessita una modellazione originale e bottom-up. Ad ogni agente viene conferito un minimo insieme di regole, tra le quali quelle riguardanti le modalità di comunicazione, di ricevere feedback, magari di riconoscere semplici pattern, avere un certo grado di scelta e di libertà, permettergli una piccola memoria, sostanzialmente di adattarsi.

Si sono usate coscientemente parole imprecise come *minimo* e *piccolo* per descrivere quantità di regole. La ragione risiede nella difficoltà intrinseca alla modellazione per CAS di stabilire a priori, per ogni diverso sistema, qual è la minima quantità di regole che un suo modello debba avere per simulare al meglio la sua natura complessa. Difficile è anche sapere con precisione i valori ottimali delle variabili del modello. Essi verranno stabiliti nel tempo grazie a continui raffinamenti. Inoltre, dipende enormemente dal tipo di sistema che si vuole modellare. È ben possibile creare modelli di sistemi con agenti molto poco intelligenti, si pensi al famoso Gioco della vita di Conway, governati da regole molto semplici. Come è anche possibile modellare sistemi

molto complessi, ad esempio CASS nei quali gli agenti sono esseri umani. Modelli i quali agenti sono quindi molto intelligenti e complessi a loro volta, e le regole che li governano copiose.

In ogni caso governa l'equilibrio. Ed esso non si ottiene soltanto risolvendo delle equazioni che ottimizzino tali parametri, per quanto sia possibile approssimarli. I valori ottimi di quantità e complessità di regole che governano ogni agente vanno migliorati progressivamente per far convergere il più possibile il modello al sistema atteso.

L'approccio alla modellazione di un CAS per mezzo di software richiede un paradigma alla programmazione particolare. Solitamente si scrive software modellando ogni aspetto del sistema e delineando tutti i comportamenti che esso debba avere, secondo un approccio top-down. Per modelli computazionali rivolti a CAS invece bisogna ragionare al contrario modellando il sistema dal basso, ovvero descrivendo i componenti singoli del sistema, i suoi agenti, dotandoli di un insieme minimale di regole comportamentali, secondo un approccio bottom-up. L'ambiente in cui essi vivono può essere ancora scritto in modalità top-down. Si lascia poi eseguire la simulazione e si osservano i suoi comportamenti, sperando che il modello manifesti segni di sviluppo, di auto organizzazione, di adattamento, tutto in maniera automatica, non preimpostata, né programmata. Si spera che il sistema modellato si sviluppi, si auto organizzi, si adatti autonomamente e generi fenomeni emergenti. Sperare perché non si hanno garanzie sulla riuscita del modello e ancor meno dell'affioramento di comportamenti emergenti. Se ci fosse certezza non si potrebbe chiamare Emergence.

Casualità

Un approccio largamente usato nella ricerca di modelli per CAS è ricorrere ad una astrazione di casualità. Alcuni processi e meccanismi vengono quindi simulati usando una certa quantità di componente aleatoria

La scelta di semplificare alcuni meccanismi complessi, assumendo che essi siano governati da funzioni stocastiche non deve spaventare. Nella sperimentazione scientifica, si è notato come l'assunzione di astrarre e semplificare certi processi dotandoli di un grado di casualità, risulta un'idea vincente. Nella maggioranza dei casi infatti, il modello sarà in grado di comportarsi in modo coerente al sistema studiato e rappresentato. Il motivo risiede nell'impossibilità di conoscere in modo deterministico e preciso ogni aspetto di sistemi in cui regna la complessità. Anche solo nello studio, ancor prima della modellazione o della simulazione, si assume che alcuni meccanismi, alcune funzioni o processi avvengano casualmente, o in un certo intervallo di probabilità, anche quando non sono intrinsecamente casuali.

In modelli computazionali applicati a CAS, l'uso di generatori di numeri pseudo casuali risulta di fondamentale importanza, per produrre alcuni aspetti celati nella complessità di questi sistemi. La scelta di quali generatori usare non è banale e anzi risulta quindi un punto critico nella modellazione.

Un'altra scelta importante da compiere da parte di chi modella è come sempre dove porre l'ago della bilancia. Stabilire il punto di equilibrio. Non si può lasciare tutto al caso, il sistema non risulterebbe fedele alla realtà, sarebbe semplicemente un grande gioco di lancio dei dadi e raccolta di risultati sconnessi, senza significato. Non è possibile nemmeno quantificare tutto, stabilire sempre ogni valore, parametro, funzione, processo in modo deterministico e perfettamente accurato. In questo estremo non staremmo rappresentando veramente un sistema complesso.

La difficoltà risiede allora nella scelta del CAS che si vuole modellare e dell'insieme minimale di elementi lasciati al caso, derivata da un accurato studio della realtà di interesse

Infine, si ricorre a statistiche nella raccolta dei risultati sperimentali. L'approccio statistico ai risultati è necessario se, alla base del modello, numerosi elementi sono stati lasciati al caso. Occorre porsi domande del tipo: con quanta probabilità avviene questo comportamento, dati questi valori iniziali? Qual è il valore atteso di questa variabile aleatoria e la sua devianza, ad un dato istante di simulazione? Con quale probabilità si manifesta un pattern o emerge dal sistema un comportamento o una intelligenza globale?

More is better

Per "*more is better*" (più sono meglio è) si intende la filosofia di modellazione che prevede di dotare il sistema di un numero di agenti sufficientemente grande affinché aumentino le probabilità di osservare fenomeni di Emergence. I CAS sono sistemi multi-agenti e nella rete di interconnessioni e dipendenze risiede parte della sua complessità. Come si può intuire, più questa rete è densa o composta da più nodi-agenti, e più è facile che emergano proprietà di complessità. Meno agenti sono presenti meno interconnessioni ci sono e di conseguenza, meno interessante apparirà il sistema, o semplicemente più raramente si manifesteranno comportamenti emergenti.

Anche qui l'equilibrio è fondamentale. Cercare e trovare il numero adatto di agenti nel sistema è uno degli obiettivi primari di chi sviluppa modelli computazionali per CS. Rispetto ad altri punti di equilibrio prima affrontati questo è il più flessibile, difatti non è veramente cruciale il numero esatto di agenti presenti nel sistema e che tale valore sia ottimale. Quel che veramente importa è trovare il numero sufficientemente elevato

affinché il modello abbia qualità, sia conforme alla teoria, rappresenti veramente il CAS modellato. "Melius abundare quam deficere".

Il limite superiore a tale valore è dato dalla potenza computazionale a disposizione e dalla memoria del computer sul quale il modello computazionale viene eseguito. Una buona stima di questo numero "*sufficientemente grande*" può quindi essere vicino al numero massimo di agenti che il computer sul quale viene eseguito il modello sia in grado di calcolare, evitando di affollare la memoria o sovraccaricare la CPU.

Maggiore è la potenza computazionale, maggiore sarà la capacità di modellare sistemi estremamente complessi. In ambito scientifico e di ricerca infatti, i modelli più evoluti fanno uso estensivo di cloud computing, calcolo distribuito e super computer per reggere il carico computazionale necessario allo sviluppo di nuove teorie e scoperte scientifiche.

Parallelismo

Una qualità ricercata nei modelli computazionali per CAS è il loro grado di parallelismo. Tale qualità risulta necessaria se si vogliono modellare sistemi "vivi", nei quali ogni agente pensa, comunica, sceglie e vive in modo autonomo. Un approccio potrebbe essere quindi modellare ogni agente come governato da un processo eseguito parallelamente a tutti gli altri. Questo aumenterebbe il livello di dinamismo del sistema e favorirebbe imprevedibilità e comportamenti emergenti. Di contro un modello così altamente parallelo richiede una grande potenza computazionale e risorse di memoria e di energia. Un'altra soluzione sarebbe quella di rendere paralleli dei macro processi e non i singoli agenti. Questi ultimi interagirebbero in sequenza. Questa soluzione è più leggera in termini di potenza computazionale richiesta e meno energivora se sviluppata al meglio. Tuttavia potrebbe non rispecchiare completamente la filosofia della natura di un CAS in cui ogni agente dovrebbe avere vita propria ed agire sì in modo interdipendente ma autonomo.

Altre soluzioni sono possibili, ma il concetto di fondo cercare di bilanciare la complessità computazionale con la complessità del modello sviluppato, cercando di utilizzare al meglio paradigmi di programmazione parallela e distribuita, simulando ciò che avviene in natura in veri CAS.

4. Il nostro modello

Presentiamo finalmente il nostro modello di sistema complesso adattivo, scopo ultimo di questo lavoro di ricerca.

Sono state valutate diverse ipotesi di sistemi da modellare per vie computazionali, alla fine si è scelto di ricorrere ad un CASS straordinario: il formicaio.

Perché questa scelta? I formicai sono sistemi complessi adattivi tra i più studiati ed osservati in natura. La facilità di reperire fonti e la copiosità di ricerche su tali sistemi ha favorito la preferenza di questo sistema rispetto ad altri. Inoltre è un sistema che si presta ad essere modellato per vie computazionali perché esprime in modo diretto un po' tutte le caratteristiche fondamentali dei CAS. Bisogna precisare che i formicai sono CASS, ovvero CAS sociali. I suoi agenti, le formiche, formano un complesso sistema sociale, ricco di interessanti proprietà, dal quale scaturisce molta della complessità e dell'adattività del sistema. Essendo un CASS, vedremo come incorpora pienamente la formulazione teorica costruita finora, anzi ci permetterà di avere nuove prospettive sui sistemi sociali non ancora profondamente esplorati.

L'esempio di riferimento: il formicaio

Lo studio di formicai si fonda su una ricerca approfondita, maturata durante la lettura di molteplici fonti. Prima fra tutte Emergence di Steven Johnson, che racconta di un caso specifico di formicaio. Altre fonti utilizzate sono state elencate nel capitolo "Sitografia".

Un formicaio è un CASS completo di tutte le caratteristiche analizzate precedentemente. Questo ci permette quindi di sfruttarlo come terreno di studi per la correttezza e completezza della teoria costruita finora, e infine come base di riferimento per la sua modellazione.

Le formiche presentano migliaia di specie differenti, ognuna con le sue peculiarità. Ci limitiamo quindi a descriverne solo gli aspetti che generalmente si applicano a tutte loro, oppure racconteremo degli aspetti tipici di una sola specie ma che si legano profondamente ad argomenti di complessità.

Una formica è un invertebrato a sei zampe, tre principali parti del corpo (capo, torace e addome) caratterizzati da un sistema nervoso piuttosto semplice.

La colonia è il gruppo di formiche che vive insieme in società, mentre per formicaio si intende sia la struttura dove esse vivono sia la colonia che vive al suo interno. La maggioranza delle specie vive sottoterra dentro una rete di sale collegate da gallerie, costruite da loro stesse.

Esse formano un sistema sociale piuttosto complesso data la relativa semplicità dei suoi elementi. In una colonia sono spesso presenti delle gerarchie e dei ruoli, chiamate caste. Una colonia è composta da formiche della stessa specie e talvolta formiche parassite anche di altre specie. In molte specie è presente una formica regina che ha il solo compito di riprodursi e deporre le uova. Ci sono poi le operaie, sempre femmine, che si distinguono in interne ed esterne. Le operaie interne si occupano di accudire le larve, di generare i maschi e nuove regine e di manutenere la struttura del formicaio. Le operaie esterne vanno in cerca di cibo nell'ambiente esterno al formicaio. Infine sono spesso presenti le formiche soldato, le quali si occupano della difesa della colonia, dell'attacco verso altre colonie, e dell'identificazione ed eliminazione dei parassiti nella colonia.

Dove risiede la complessità di un formicaio? Nella fitta rete di formiche comunicanti che lo popolano. Un formicaio segue un sistema di regole bottom-up, manifesta una serie di fenomeni emergenti dal sistema sociale, è fortemente adattivo e capace di auto organizzarsi.

Ad esempio, la scelta della divisione in caste è principalmente in mano alle singole operaie che si occupano della prole. Esse sanno che un tipo di alimentazione o un metodo specifico di supporto alla crescita delle larve produce un tipo specifico di forma fisica che stabilisce la casta della futura formica adulta. La scelta (dell'insieme di operazioni di crescita della prole che causa la selezione delle caste) spetta ad ogni operaia autonomamente. Nessuna riceve ordini da altre. Eppure, il sistema si autogoverna e riesce a bilanciare *quasi* perfettamente il numero di maschi, di operaie, di soldato e la presenza o meno della regina fecondata. Come avviene questo? Grazie ai loro meccanismi di comunicazione, primo fra tutti per via dei feromoni. Le formiche hanno uno sviluppato senso dell'olfatto e sono altamente sensibili ai feromoni. Ogni comunicazione viene cifrata in una sequenza precisa di feromoni. Basta quindi secernere un certo tipo e quantità di feromoni per comunicare un certo tipo di informazioni. Esempi di informazioni "codificabili" sono l'età e la fecondità della regina, il numero di operaie incontrate, la distanza da una fonte di cibo, la presenza di un parassita o di un agente patogeno o di un predatore. Ogni formica rilascia feromoni specifici per il tipo di informazioni che vuole passare al formicaio. Ognuna di loro dà e riceve continuamente feedback delle informazioni che essa sparge nel formicaio ed all'esterno di esso. L'insieme delle informazioni ricevute è in grado di ritrasmetterle alle altre. Così ogni formica è in grado di elaborare le informazioni ricevute, imparare da esse e ritrasmetterle a sua volta per il bene della colonia. L'insieme di queste informazioni locali che viaggiano nel formicaio stabilisce tutte le regole globali sotto le quali esso vive. Regole che sono quindi auto prodotte in modo condiviso e distribuito da tutte le formiche.

Quali comportamenti emergenti sono stati osservati nelle ricerche fatte? Si potrebbero fare molti esempi. S. Johnson descrive nel suo libro lo studio di un formicaio in laboratorio, confinato in una teca di vetro della dimensione necessaria al formicaio,

con risorse di cibo a sufficienza per il sostentamento della colonia. Si è potuto osservare, tra le molteplici scoperte, la capacità del formicaio di sistemarsi in modo sempre più ottimale allo scorrere del tempo. Nel percorso evolutivo del formicaio si è notato come le formiche si organizzassero per disporre in punti precisi la locazione del "cimitero" e del "centro rifiuti". Nel primo trasportavano le formiche morte, nell'altro i rifiuti e gli scarti del formicaio. È stata quindi calcolata la distanza tra le due zone e l'entrata al formicaio. Il pattern sorprendente emerso da questa scelta è che i tre punti erano molto prossimi ai vertici di un triangolo equilatero. La distanza tra cimitero, centro raccolta rifiuti ed ingresso formicaio era quasi perfettamente la stessa una dall'altra. È come se le formiche fossero state in grado di risolvere semplici problemi di geometria. Eppure, nessuna di loro sa farlo autonomamente, ma il sistema ne è in grado. Si è intuito e verificato come la rete fitta di comunicazioni dinamiche e non lineari tra formiche sia stata determinante affinché una informazione attraversasse tutto il formicaio. Quest'ultimo, come fosse un sistema composto da un unico componente, nel tempo, si è adattato efficacemente al suo ambiente, imparando come la disposizione dei tre luoghi in modo equidistante fosse ottimale. Non è tutto. I tre punti non solo formano un triangolo equilatero, sono anche posti ai limiti della teca di vetro, che rappresenta lo spazio di sviluppo del sistema. Questo vuol dire che la colonia non ha solo ottimizzato un problema geometrico astratto, ma è stata in grado di ottimizzarlo nel suo specifico ambiente di sviluppo prefissato. Tale comportamento ha permesso di comprendere il grande senso di adattabilità del sistema. Non solo, la scelta delle tre posizioni nello spazio non è avvenuta da un giorno all'altro. La nuova conformazione, il pattern emerso, ha attraversato varie generazioni di formiche per crearsi. Il sistema si è evoluto, anche in funzione del suo ambiente e delle informazioni ricevute e quest'ultime sono sopravvissute al passaggio generazionale. È come se il sistema possedesse una memoria accessibile a prescindere dalla vita e dalla morte delle singole formiche. Questa memoria permette la creazione di uno storico che risulta essenziale per la colonia nelle scelte che essa prende. Grazie a questa storia ogni singola formica, agente del sistema, sa trarne informazioni, compiere scelte migliori, dare feedback alle sue vicine o a quelle che incontra nel suo percorso e contribuire indirettamente alla scelta finale del posizionamento ottimale dei tre luoghi. Osservando una singola formica sarebbe impossibile avere questa visione d'insieme e scorgere questi pattern e altri comportamenti emergenti. Una singola formica compie scelte locali in funzione della sua casta, delle informazioni che ha e che riceve. Non fa calcoli, non impone scelte alle altre. Non ci sono direttori o capi.

Sono stati poi osservati comportamenti di "tutoraggio" di formiche anziane verso quelle più giovani: formiche che insegnano alle altre a raggiungere la fonte di cibo trovata. Tuttavia, anche questo rapporto di ruoli maestro-allievo è un processo stabilito localmente ed internamente alla colonia. Non esiste una scuola, un governo che stabilisce tali ruoli. In questo caso si tratta di una scelta tra due singole formiche. Eppure, questa piccola scelta locale di ruoli in una singola coppia permette l'effetto, non lineare, di garantire il sostentamento e la sopravvivenza all'intera colonia. La

formica “allievo” diventerà successivamente “maestro” per poter insegnare ad altre a raggiungere il cibo, l’informazione si propaga e la colonia continua a vivere.

Nonostante l’apparente ottimalità del formicaio osservato in laboratorio, esso rimane un caso particolare perché confinato in un ambiente statico. Nella realtà l’ambiente è altamente dinamico. In natura si è osservato, infatti, come non ci sia fine al processo adattivo e di ricerca dello stato ottimo da parte del formicaio. Lo stato ottimo presente, non lo è più dopo un certo intervallo di tempo. Si è osservato che molte specie adottano strategie volte a prevedere l’evoluzione del proprio ambiente preparandosi in anticipo, è qui che avviene il mistero dell’anticipazione. Rimane ancora oscuro ai ricercatori come ciò avvenga.

Esistono alcune specie di formiche in grado di aggregarsi per uno scopo ben preciso, come creare delle forme usando i loro stessi corpi, ad esempio un ponte tra un punto ed un altro o delle “barche”, al fine di attraversare piccoli corsi d’acqua. Tutto ciò funziona e avviene senza ingegneri o capi cantiere, ogni singola formica comunica le proprie scelte alle altre e di comune accordo stabiliscono la forma e la modalità di aggregazione per rispondere ad un bisogno comune. Questo comportamento si è riscontrato in pochissime altre specie animali, nelle api ad esempio.

Altre specie di formiche si dimostrano altamente aggressive ed entrano facilmente in competizione con i formicai limitrofi. Sono state osservate semplici strategie di guerra, di razzie tra colonie diverse e addirittura schiavismo di formiche nemiche catturate. Questi comportamenti fanno parte della strategia a lungo termine messa in atto dalla colonia, al fine di raggiungere i propri obiettivi.

L’insieme di esempi e conseguenze tratte dallo studio e dalla ricerca sui formicai riportati finora, crediamo sia sufficiente per rafforzare la teoria costruita affiancandola ad un esempio valido in natura di CASS. Abbiamo potuto infatti vedere in pratica dei meccanismi di emergence, di pattern chiari e visibili, di comportamenti di aggregazione di organizzazione e adattamento.

Siamo pronti per passare alla pratica.

Descrizione del software e del modello computazionale

Daremo una descrizione dei componenti principali, delle tecniche di programmazione più importanti utilizzate, delle astrazioni che abbiamo scelto di compiere sul sistema e di come il modello rimane fedele al sistema di riferimento.

Il software progettato è scritto in Java 17, è composto da una GUI e da tre modelli computazionali, che sono dei CAS. La GUI si occupa di mostrare a schermo il comportamento e l'evoluzione del modello sottostante. Tramite un menù è possibile scegliere quale dei tre modelli visualizzare. Il primo (in ordine crescente di complessità) è il celebre Gioco della Vita ideato da Conway, il secondo un Social Game System, il terzo un formicaio.

In questo lavoro entreremo nel dettaglio solo del terzo modello essendo il più interessante e complesso dei tre. Nel capitolo "Sitografia" è presente il GitHub del progetto con tutti e tre i modelli implementati.

Il modello del formicaio prende forma dallo studio della teoria raccolta nel paragrafo precedente. Vediamo quindi come lo abbiamo costruito a partire da questo sistema formicaio.

Il modello è sviluppato secondo un paradigma orientato agli oggetti e bottom-up. Ovvero nel delinearne le regole costitutive, è stato enfatizzato il comportamento di ogni singola formica. Poche sono le regole che governano tutto il sistema, la maggioranza di esse riguardano la gestione tecnica del codice oppure governano gli elementi esogeni del sistema, come le sorgenti di cibo.

Una classe, istanziata sempre da un solo oggetto, si occupa della gestione del modello in generale, di invocare l'esecuzione del codice relativo di ogni componente del sistema, di gestire il corretto accesso alle risorse evitando possibili "race conditions" tra thread differenti (principalmente tra il thread della GUI e quello del modello sottostante) e infine si occupa di creare e gestire le statistiche del modello.

Il modello è fondamentalmente *asincrono e sequenziale*. Ogni formica è rappresentata da un oggetto Java, la classe generale che si occupa di gestire il modello, invoca sequenzialmente ogni oggetto formica ancora vivo e gli fa eseguire il suo codice. Una dopo l'altra, ogni formica viva nel sistema, viene invocata ed esegue delle azioni. In seguito diremo quali azioni può svolgere ciascuna formica in ogni turno. Ogni giorno, tutte le formiche vive si attivano ed eseguono delle azioni. Le formiche vengono invocate sequenzialmente in ordine casuale ad ogni turno, così da non favorire direttamente nessuna formica rispetto ad un'altra. Nel corso del tempo ogni formica

ha probabilità equa di essere scelta per prima o per ultima ogni giorno. Inoltre questa classe si occupa di creare fonti di cibo sulla griglia e di tenere aggiornate le strutture dati che conoscono in ogni momento, per ogni cella sulla griglia, quale elemento del modello è presente.

L'ambiente del sistema è modellato con una griglia bidimensionale 60x60 (le dimensioni sono modificabili), che rappresenta l'ambiente dove il sistema si sviluppa e con il quale interagisce, ne delimita le sue dimensioni e le sue interazioni possibili.

Nel mondo bidimensionale di questo modello convivono due principali attori, i componenti endogeni del sistema formicaio (formiche, nido e feromoni) e le fonti di cibo che modellano quindi la componente esogena al sistema, quella con il quale il sistema interagisce esternamente ad esso. Tutti e quattro gli elementi vengono mostrati sulla griglia in ogni momento, ma su ogni cella della griglia è presente al più un solo elemento in ogni istante t.

Il modello è fondamentalmente *asincrono e sequenziale*. Ogni formica è rappresentata da un oggetto Java, la classe generale che si occupa di gestire il modello, invoca sequenzialmente ogni oggetto formica ancora vivo e le fa eseguire il suo codice. Una dopo l'altra, le formiche vive nel sistema, vengono invocate, ogni giorno in ordine casuale con probabilità uniforme, così da non favorire nessuna formica rispetto ad un'altra, ed eseguono delle azioni (in seguito diremo quali). Inoltre questa classe si occupa di creare fonti di cibo sulla griglia e di tenere aggiornate le strutture dati per sapere, in tempo reale, quale elemento del modello è presente in ogni cella della griglia.

Le **formiche** vengono visualizzate come semplici quadratini di colore verde scuro. Per semplificare il sistema, ove possibile, si è rinunciato a modellare diverse caste di formiche. Esiste quindi un solo tipo di formica nel modello, svolgente le funzioni caratteristiche di una operaia esterna ma anche, usando una astrazione, di alcune funzioni svolte dalle operaie interne, tra cui l'accudimento della prole.

Ogni formica ha due stomaci come nella realtà, uno pubblico ed uno privato. Ogni volta che raccoglie cibo, lo pone dapprima nello stomaco pubblico. Questo nome deriva dalla sua funzione di supporto alle altre formiche della colonia. Se una formica A incontra una formica B avente nel suo stomaco pubblico minor cibo della formica A, quest'ultima può decidere di passare parte del cibo contenuto nel proprio stomaco pubblico allo stomaco pubblico della formica B. Questa operazione si chiama trofallassi. Nel modello le formiche eseguono trofallassi con una probabilità

proporzionale alla differenza di cibo esistente nel momento dell'incontro tra due di esse. Maggiore è la differenza, maggiore è la possibilità che avvenga trofallassi. Lo stomaco privato, come si può intuire dal nome, serve alla formica per nutrire soltanto sé stessa.

Ogni formica è caratterizzata da uno stato principale che ne stabilisce l'intenzione o lo scopo chiamato "toTheNest". Se la formica presenta questo stato tenderà a muoversi sul percorso che le permette di raggiungere il più velocemente possibile il proprio nido. Se invece non è in questo stato avrà l'atteggiamento opposto, quello di tendere ad esplorare l'ambiente circostante.

Allo scopo di modellare il sistema nervoso della formica, la sua capacità di reagire agli stimoli, di interagire, di comunicare e di compiere semplici scelte, vengono usati metodi diversi che svolgono alcune funzioni chiave. Alcuni di questi metodi utilizzano semplici algoritmi di intelligenza artificiale adattati ad-hoc per questo modello. Uno degli obiettivi da raggiungere è stabilire un equilibrio, tra una formica "semplice", con sole regole di azione-reazione in risposta a stimoli esterni, ed il suo opposto, cioè una formica "onnisciente", capace di agire su tutti i processi e gli elementi del sistema.

In natura ogni formica è capace di riconoscere l'odore del proprio nido fino a 100 metri di distanza e di stabilire quasi perfettamente il percorso ottimale per raggiungerlo. Questa capacità è stata modellata facendo in modo che ogni giorno la formica aggiorni un array di strutture dati chiamate direzioni. Le direzioni sono le 4 cardinali: nord, sud, est ed ovest; e le loro 4 combinazioni: NO, N, NE, E, SE, S, SO, O. Ognuna rappresenta la direzione da seguire per raggiungere una delle 8 caselle intorno alla formica, ogni casella adiacente ha quindi una sola direzione associata.

Infine, ogni formica ha un proprio codice genetico che viene modellato come l'insieme dei valori di otto differenti variabili, otto caratteristiche che descriveremo da qui in poi. Quando la formica depone uova passerà alla prole parte del proprio codice genetico così da simulare una riproduzione che tende ad evolvere il sistema globale. Vedremo anche questo nel dettaglio più avanti.

Ogni giorno una formica esegue in sequenza le seguenti azioni:

1. Trasferire il cibo dallo stomaco comune a quello privato.
2. Mangiare il cibo contenuto nello stomaco privato
3. Orientarsi
4. Interagire con altri elementi del sistema intorno a sé
5. Scegliere cosa fare
6. Scegliere se raggiungere del cibo vicino
7. Muoversi
8. Invecchiare

Analizziamo queste attività.

La *prima azione* serve a trasferire una piccola parte del cibo contenuto nello stomaco pubblico verso lo stomaco privato, questo avviene anche nella realtà per permettere alla formica di nutrirsi dallo stomaco privato.

La *seconda azione* è quindi quella di consumare una parte del cibo nello stomaco privato. Se lo stomaco è completamente vuoto la formica può entrare in uno stato di fame acuta che, se prolungato nel tempo, la porta progressivamente a diminuire le sue chance di sopravvivenza.

La *terza azione* serve per l'orientamento. Ogni formica viva in un tempo "t" occupa una cella sulla griglia. Ogni cella ne ha otto intorno a sé codificate con i nomi delle otto direzioni cardinali. Ogni giorno la formica ricalcola le distanze di ognuna delle otto caselle intorno a sé e le ordina dalla più vicina alla più lontana. Le prossime azioni avranno bisogno di sapere le otto direzioni ordinate.

La *quarta azione* è quella più ricca di sotto azioni. La formica controlla se, nelle otto caselle che la circondano, è presente un altro agente del sistema e vi interagisce in base alla sua natura.

- Se incontra un'altra formica verifica l'esistenza delle condizioni necessarie alla trofallassi;
- Se trova una fonte di cibo ne raccoglie una parte nel suo stomaco pubblico ed inizia a rilasciare feromoni sul suo percorso (spiegheremo più avanti questo fenomeno)
- Se raggiunge il nido del formicaio, può svolgere una delle due azioni opposte: raccogliere un po' del cibo conservato nel nido se la formica ne possiede poco nei suoi stomaci; oppure dare una parte del cibo che sta trasportando alle riserve del formicaio se sta trasportando cibo a sufficienza per sé stessa e per altri. Poi se ha depositato del cibo la formica depone due uova nel nido, passandogli parte del proprio codice genetico (in seguito spiegheremo l'algoritmo genetico che se ne occupa). Comunque sia alla fine cambia il suo stato principale "toTheNest" a falso, indicando di essere in tendenza esplorativa.
- Se individua tracce di feromoni, cerca quella ad intensità minore intorno a sé, che chiamiamo "leph".

La *quinta azione* è quella più complessa. In questa fase la formica sceglie la direzione di movimento in base a parametri quali: il suo stato principale, gli incontri avuti e le tracce di feromone incontrate.

Essendo la funzione più importante di ogni formica, ne presentiamo il codice commentato in ogni passaggio.

```
mainly decide the direction to follow based on the information gathered until now
private <E> void decide() {
    // First key AI logic of an ant:
        // if I am starving or have little food go to you nest to feed
        // at the opposite if I have plenty of food with me, it is useless to continue finding food in the ambient so come back to the nest
        // if I have just enough quantity of food to go exploring then I will prefer to go outside and explore the ambient for food
    boolean found = false;
    // remember that maxStomachCapacity it's the maximum capacity of a single stomach
    // if I decided time ago to go to the nest don't change that decision until you arrive to the nest
    if (roaming > minRoaming && (toTheNest || ((stomachSum < maxStomachCapacity * 0.4)  || (stomachSum > maxStomachCapacity * 1.2)))) {
        double p = random_seed.nextDouble();
        int start;
        if (p < 0.6) start = 0;         // with a P of 60% choose to start to search a direction to follow from nestDirections[0] which correspond to the closest path to the nest
        else if (p < 0.9) start = 1;    // else with a P of 30% choose nestDirections[1] which correspond to the second-closest path to the nest
        else start = 2;                 // else with a P of 10% choose to start from nestDirections[2] which correspond to the third-closest path to the nest
        for (int i = start; i < start + 3; i++) {  // iterate the 3 possible directions to find the first one free starting from the one chosen before as a starting point
            if (!found) {
                translateDirInPos(nestDirections.get(i % 3));  // updates nextY and nextX
                E el = whoIsThere(nextY, nextX);
                if (el == null || el.getClass() == Pheromone.class) {  // if the direction is already occupied then skip
                    chosenDir = nestDirections.get(i % 3);  // change the direction to follow
                    found = true;
                    toTheNest = true;  // change main state
                }
            }
        }
    }
```

Parte prima: ogni formica decide se cambiare il proprio stato principale “toTheNest” a vero, la conseguenza sarà quella di orientare la formica a compiere azioni che la dirigono verso il nido. Questa decisione si basa su tre fattori:

1. Se ha superato il numero minimo di turni passati ad esplorare. Tale numero è uno degli otto geni che compongono il DNA di una formica;
2. Se precedentemente ha già scelto di andare verso il nido;
3. Sulla quantità di cibo presente nei suoi stomaci: se molto bassa o molto alta la formica vorrà tornare al nido.

Se la prima condizione ed almeno una delle altre due sono vere allora, la formica, cambierà stato “toTheNest” a vero, e sceglierà una direzione da seguire, tra le tre più dirette al nido. Le direzioni sono ordinate sempre dalla più vicina alla più distante dalla funzione di orientamento. Ogni formica sceglie una delle tre direzioni più vicine con probabilità crescente, la direzione che porta alla casella più vicina avrà la probabilità più alta di essere scelta. Questo servirà per modellare la capacità di una formica di intraprendere il percorso più veloce possibile dalla propria posizione al nido.

```
    // Second key AI logic of an ant:
        // if I found a strong pheromone trail around me go to the lightest one, hopefully in the opposite direction of the strongest one
        // this may lead me to a food's source
    if (pheromoneCounter < 5) { // if there are too many pheromones around you, you'll get very confused, so ignore them
        if (leph != null) {  // if you have the Lightest Encountered PHeromone (see pheromoneIntercation())
            if (Math.random() < ((Pheromone.maxStrength - leph.getStrength()) / (double) Pheromone.maxStrength) * 1.2) { // with a P proportional to the strength of leph
                Direction desirableDir = translatePosInDir(leph.yPos, leph.xPos);
                if (toTheNest) {  // follow it ONLY if it's in one of the three squares around you closest to the nest
                    for (int i = 0; i < 3; i++) {
                        if (nestDirections.get(i) == desirableDir) {
                            chosenDir = desirableDir;
                            found = true;
                            break;
                        }
                    }
                } else {  // follow it ONLY if it's in one of the five squares around you farthest from the nest
                    boolean ok = true;
                    for (int i = 0; i < 3; i++) {
                        if (nestDirections.get(i) == desirableDir) {
                            ok = false;
                            break;
                        }
                    }
                    if (ok) {
                        chosenDir = desirableDir;
                        found = true;
                    }
                }

            }
        }
    }
    if (found) {  // reset some flags, see movement()
        onARandomPath = false;
        countDir = 0;
    }
}
```

Parte *seconda*: in funzione dello stato principale la formica sceglie se seguire una delle tracce di feromone tra quelle intorno a sé.

Se il numero di feromoni adiacenti supera 5 (su 8) la formica evita di sceglierne uno per la difficoltà nello stabilire il migliore, questo serve al modello perché le formiche non si incastrino in loop di feromoni ciclici senza mai tendere ad esplorarne altri lontani da sé. La logica qui usata è quella di bilanciare *exploitation* ed *exploration,* punti focali di molti algoritmi di intelligenza artificiale. Se ci fosse solo exploitation (traducibile con inferenza) la formica non esplorerebbe mai e tenderebbe a compiere movimenti ripetitivi, uguali e ciclici. Con solo exploration (esplorazione) il sistema tende al caotico e niente di veramente interessante ne emergerebbe.

Si può notare come questa seconda parte dell'algoritmo ruota fondamentalmente attorno al concetto di "*leph*" che sta per Lightest Encountered PHeromone. Il leph rappresenta la traccia di feromone più debole tra quelle attorno. Possiamo pensare a leph come un valore su cui ogni formica fa inferenza nel decidere la direzione dove muoversi. L'algoritmo inoltre esprime una preferenza in questo valore di inferenza. Se si è nello stato esplorativo ("toTheNest" = falso) la formica si muove verso il leph solo se si trova su una delle cinque (su otto) posizioni più *lontane* dal nido, tra quelle intorno a sé. Viceversa se si sta dirigendo verso il nido ("toTheNest" = vero) la formica si muove verso il leph solo se si trova su una delle tre (su otto) posizioni più *vicine* dal nido, tra quelle intorno a sé.

La *sesta azione* modella l'olfatto di una formica. Ogni formica in natura è in grado di percepire una fonte di cibo a molti metri di distanza. Questo comportamento viene modellato dotando la formica della capacità di percepire cibo ad una distanza massima di tre caselle. Questo significa che una fonte di cibo posizionata all'interno di un quadrato di lato 7 caselle (un 7x7 immaginando la formica al centro ed un raggio di 3 caselle nelle 8 direzioni) sarà con grande probabilità oggetto di un cambio di direzione da parte della formica che lo scopre. La probabilità aumenta se la formica è in fase esplorativa ("toTheNest" = vero). Questa azione sovrascrive la quinta azione, perché percepire cibo ha maggiore priorità rispetto a percepire una traccia di feromoni.

La *settima azione* si occupa del movimento di una formica. Se la formica ha scelto una direzione nelle azioni precedenti (seguendo una traccia di feromoni o una fonte di cibo) allora invoca il metodo che si occupa di spostare la formica nella casella adiacente lungo la direzione scelta. Inoltre la formica mantiene la direzione scelta per molti turni finché non sceglie di cambiarla o trova uno ostacolo di qualsiasi tipo, in questo caso la formica ricalcola una nuova posizione libera dove dirigersi.

Come funziona il ricalcolo del percorso?

Una formica stabilisce una posizione dove muoversi in funzione di una serie di valori. Dipende prima di tutto dalla posizione della formica nell'istante presente.

Una funzione calcola la probabilità di scegliere una delle otto direzioni in base alla distanza delle celle circostanti. È possibile visualizzare la funzione su Desmos al secondo link lasciato nel capitolo "Sitografia".

Sia dMax la distanza massima dal nido in cui una formica può trovarsi, sia halfDmax la metà di questo valore, halfDmax rappresenta un raggio che disegna un cerchio intorno al nido. Se la formica è fuori da questo cerchio, in una posizione a distanza maggiore della metà del raggio massimo, avrà più voglia di tornare verso il nido, con probabilità crescente in funzione della distanza da quest'ultimo. Viceversa se si trova all'interno del cerchio di raggio halfDmax avrà più voglia di allontanarsi dal nido con probabilità crescente alla distanza da esso. Sia inoltre "GUI.DIMENSION" il numero di celle sulla griglia e sia "i" l'indice dell'array "closestNestDistance", l'array che mantiene le distanze delle otto celle adiacenti in ordine di distanza crescente. Chiamiamo "x" la nostra variabile che equivale a closestNestDistance.get(i) che a sua volta equivale alla distanza della cella in posizione i-esima nell'array. La seguente funzione per parti calcola una probabilità.

$$\begin{cases} f_1(x) = -\dfrac{(\mathrm{x} - \mathrm{halfDmax}) * (\mathrm{i}+1)}{\mathrm{halfDmax} * 2\ln(GUI.DIMENSION)} + \dfrac{1}{8}, & 0 \le \mathrm{x} < \mathrm{halfDmax} \\ f_2(x) = \dfrac{(\mathrm{x} - \mathrm{halfDmax}) * (8-i)}{\mathrm{halfDmax} * 2\ln(GUI.DIMENSION)} + \dfrac{1}{8}, & \mathrm{halfDmax} \le \mathrm{x} \le \mathrm{dMax} \end{cases}$$

Infatti: $0 \leq f_{1\,o\,2}(x) \leq 1$. Questa probabilità, che dipende quindi dalla distanza di una cella dal nido esprime la possibilità con la quale la formica sceglie di seguire la direzione associata a x.

Esponiamo una parte del codice che si occupa di implementare questa funzione. Il metodo Java calcola p con $p = f(x)$ ed estrae un valore casuale tra 0 ed 1. Se il valore estratto è minore di p controlla se la direzione associata è libera. Se il controllo è positivo la formica ha calcolato una nuova direzione da seguire. Si può notare che il calcolo comincia da i = 7 ovvero dalla cella a maggiore distanza dal nido, questo perché siamo nel caso in cui la formica è all'interno del cerchio di raggio halfDmax, perciò dà la priorità alla cella a distanza massima dal centro. L'altra parte del metodo implementa $f_2(x)$, iniziando da i = 0, quindi dalla cella a distanza minore, essendo la formica oltre il cerchio di raggio halfDmax.

```
compute a random direction to follow. The logic is to give better chance to go to a direction depending
on the distance of your position from the nest. That is greater the distance the more you would want to
go towards the nest, closer the distance the more you would want to go away from it.
private <E> void findRandomDirection() {
    random_seed = new Random();
    E obstacle;
    double halfDMax = dMax / 2;  // dMax is the maximum distance an ant can be in the cell on the grid farthest from the nest
    boolean found = false;

    // inside the halfway circle of ray = halfDMax
    if (closestNestDistances.get(4) < halfDMax) {
        for (int i = 7; i >= 0; i--) {  // search through the 8 directions
            // linear function that depends on the value of closestNestDistances.get(i)
            // also depends on GUI.DIMENSION = the number fo cells on the grid
            // and on halfDmax the maximum ray from the centre, halved
            // starting from i = 7 where closestNestDistances.get(7) correspond to the cell farthest from the nest
            // this will have the greatest Probability of being chosen
            double p = -((closestNestDistances.get(i) - halfDMax) / (2 * Math.log(GUI.DIMENSION) * (halfDMax) / (i + 1))) + 1 / 8.0;
            System.out.println("pIN" + p);
            Direction possibleDir = nestDirections.get(i);
            translateDirInPos(possibleDir);  // updates nextY and nextX
            obstacle = whoIsThere(nextY, nextX);
            if (obstacle == null || obstacle.getClass() == Pheromone.class) {  // cell is free
                if (notOpposite(possibleDir)) {  // it isn't the opposite direction to the current one
                    if (random_seed.nextDouble() < p) {  // choose this direction with a Probability of p
                        if (inBounds(nextY, nextX)) {    // if inside the grid
                            chosenDir = possibleDir;
                            countDir = 0;
                            onARandomPath = true;
                            found = true;
                            break;
                        }
                    }
                }
            }
        }
```

La ultima e *ottava azione* si occupa semplicemente di aumentare l'età della formica. Ogni formica ha una aspettativa di vita di cento giorni se, nell'arco della sua vita, non si trova mai in stato di fame acuta.

Le **fonti di cibo** vengono visualizzate sulla GUI da quadratini colorati in giallo che vengono generati casualmente sulla griglia.

Al tempo t = 0 vengono generati un numero di fonti di cibo sulla griglia compreso tra due valori: minimal e maximal, che sono in funzione del numero totale di celle sulla griglia. Un generatore casuale di numeri pseudorandomici a distribuzione uniforme sceglie un valore tra minimal e maximal che stabilisce il numero di fonti di cibo iniziali.

La generazione di fonti di cibo sulla griglia avviene anch'esso con un generatore casuale a distribuzione uniforme.

Ogni giorno successivo al primo, con una probabilità del 5%, viene generata casualmente una nuova fonte di cibo sulla griglia, se la posizione scelta a caso, è libera.

Il **nido** viene modellato con quattro celle inamovibili poste al centro della griglia. Ogni formica conosce la posizione del suo nido e sa come raggiungerlo. Il nido svolge due funzioni chiave:

1. Custodisce le risorse di cibo del formicaio.
2. Genera nuove formiche.

La generazione di nuove formiche segue l'idea di un algoritmo genetico.

Ogni formica ha un codice genetico formato da otto attributi:

1. Il numero di attributi (tra questi otto) da passare ai figli;
2. Ogni quanti giorni cambiare direzione se si è su un percorso scelto a caso da "findRandomDirection()" (codice della funzione mostrato precedentemente);
3. Quanti giorni consecutivi rilasciare feromoni sul proprio percorso;
4. Qual è l'intensità dei feromoni rilasciati;
5. Qual è la capacità massima di uno stomaco (entrambi hanno la stessa capacità);
6. Quanto cibo mangiare al giorno;
7. Quanto cibo trasferire dallo stomaco pubblico a quello privato ogni giorno;
8. Qual è il numero minimo di turni consecutivi in cui si è in stato esplorativo.

Il valore di ognuno di questi valori viene calcolato tramite funzioni simili tra loro che sono quasi tutte proporzionali al logaritmo di GUI.DIMENSION, che ricordiamo essere il numero totale di celle sulla griglia.

Anche il nido possiede un array che contiene questi otto valori.

La dinamica di riproduzione del formicaio viene implementata da questo algoritmo:

```
The method that implements a version of a generic genetic algorithm. The method sets the genetic code
to pass to newborn ants that is the fusion of two parts. The first part of the genetic code comes directly
from the father ant. The second part is from the genetic code already present in the nest, which is a
contribution of all the ants in the history that have transmitted its genetic code to the nest
Params: attributes – the genetic code of the father ant, which is made of 8 attributes
```

```
void transmitGenetics(ArrayList<Double> attributes) {
    // do some genetic algorithm code
    // with the crossover value "nTraitsToTransmit" and a random point to start copying
    Random r = new Random();
    int start = r.nextInt( origin: 0,  bound: 8);  // where to start copying genetics values
    for (int i = start; i < start + 8; i++) {
        if (i - start < attributes.get(0)) {    // nTraitsToTransmit between 1 and 8
            antAttributes.set(i % 8, attributes.get(i % 8));
        }
        // if it's the first ant than get its whole genetic code
        else if (antAttributes.get(i % 8) == null) {
            antAttributes.set(i % 8, attributes.get(i % 8));
        }
    }
}
```

Il metodo implementa una versione di algoritmo genetico con molti padri ed un figlio. Una formica che arriva al nido trasportando una quantità di cibo sufficientemente alta ne darà una parte al nido e poi invocherà questo metodo che le permette di trasmettere una parte del suo codice genetico contenuto in "attributes" al nido. Il nido stabilisce un punto a caso da cui cominciare a copiare il DNA della formica padre che è arrivata al nido, simulando il lavoro del DNA polimerasi che avviene in natura. Copierà tanti attributi quanto il valore del primo, ovvero di attributes.get(0). Esso implementa il valore di "*crossover*" di un algoritmo genetico. La prima formica in assoluto darà tutto il suo codice genetico al nido. Il figlio così creato sarà la sua copia esatta. Il secondo padre darà una parte del suo codice genetico. Il figlio così creato avrà una parte del codice genetico del primo padre ed una parte del secondo. Il terzo padre contribuirà dando una parte del suo DNA che andrà a sovrascrivere una parte di quello già presente nel nido e così via. L'n-esimo figlio avrà un codice genetico che è il contributo dei $k \leq n$ padri che hanno rilasciato una parte del DNA al nido sovrascrivendo quello già presente con il loro.

Questo algoritmo modella la riproduzione delle formiche che in natura avviene in modi più complessi ma questa astrazione vedremo essere ricca di interessanti conseguenze.

Infine gli ultimi elementi del modello sono le tracce di **feromoni,** il mezzo di comunicazione principale utilizzato dalle formiche in natura. Sono sostanze chimiche che secernono le formiche per comunicare. Abbiamo modellato solo un tipo di feromoni: quelli che indicano un percorso da seguire, li chiameremo feromoni di traccia o semplicemente feromoni. Ogni formica, quando incontra una fonte di cibo, decide di lasciare da quel posto una traccia di feromoni sul proprio percorso. Ogni volta che si muove, da ora in avanti, rilascia un feromone nella posizione da dove veniva. Questo permette alle altre formiche di seguire la traccia lasciata, insegnandole a raggiungere quella fonte di cibo.

Ogni formica, da quando incontra una fonte di cibo, rilascia feromoni sul suo percorso per un numero di giorni che dipende dal terzo attributo del codice genetico. Ogni feromone ha una sua intensità iniziale che decresce di uno ogni giorno. Quando arriva a zero si disperde nell'ambiente e nessuna formica è più in grado di percepirlo. Il quarto attributo del codice genetico della formica indica l'intensità iniziale dei feromoni che rilascia.

Quando una formica passa sopra una traccia di feromoni mentre sta rilasciando essa stessa feromoni sul suo percorso, rafforzerà il feromone già presente con il suo. Se le formiche rafforzano la traccia di feromoni che incontrano perché però si dirigono verso quella ad intensità minore?

Ogni formica mentre si muove rilascia feromoni sul percorso a partire da una fonte di cibo. Essa si muove da una casella ad un'altra adiacente ogni giorno che passa nella simulazione. Inoltre ogni giorno tutte le tracce di feromoni sulla griglia diminuiscono di intensità di uno. Conseguenza di questi tre fenomeni è che le formiche creano delle tracce a intensità crescente dalla fonte di cibo da dove provengono alla posizione in cui si trovano attualmente.

Quindi la strategia migliore è quella di percorrere la traccia di feromoni a ritroso nel verso opposto a quello della formica che l'ha generato. Cercare il feromone più leggero intorno a sé in ogni turno induce la formica a seguire una traccia in ordine decrescente di intensità, sperando così di arrivare alla fonte di cibo da cui è partita.

Volendo sintetizzare abbiamo tralasciato di approfondire molte parti del modello, molti algoritmi, funzioni e componenti tecniche del software. Nonostante ciò tutti gli elementi cardine del modello sono stati ampiamente analizzati e descritti e talvolta anche accompagnati da pezzi di codice che fanno vedere la loro implementazione. Possiamo ora passare all'analisi dei risultati e dei componenti emergenti dal modello.

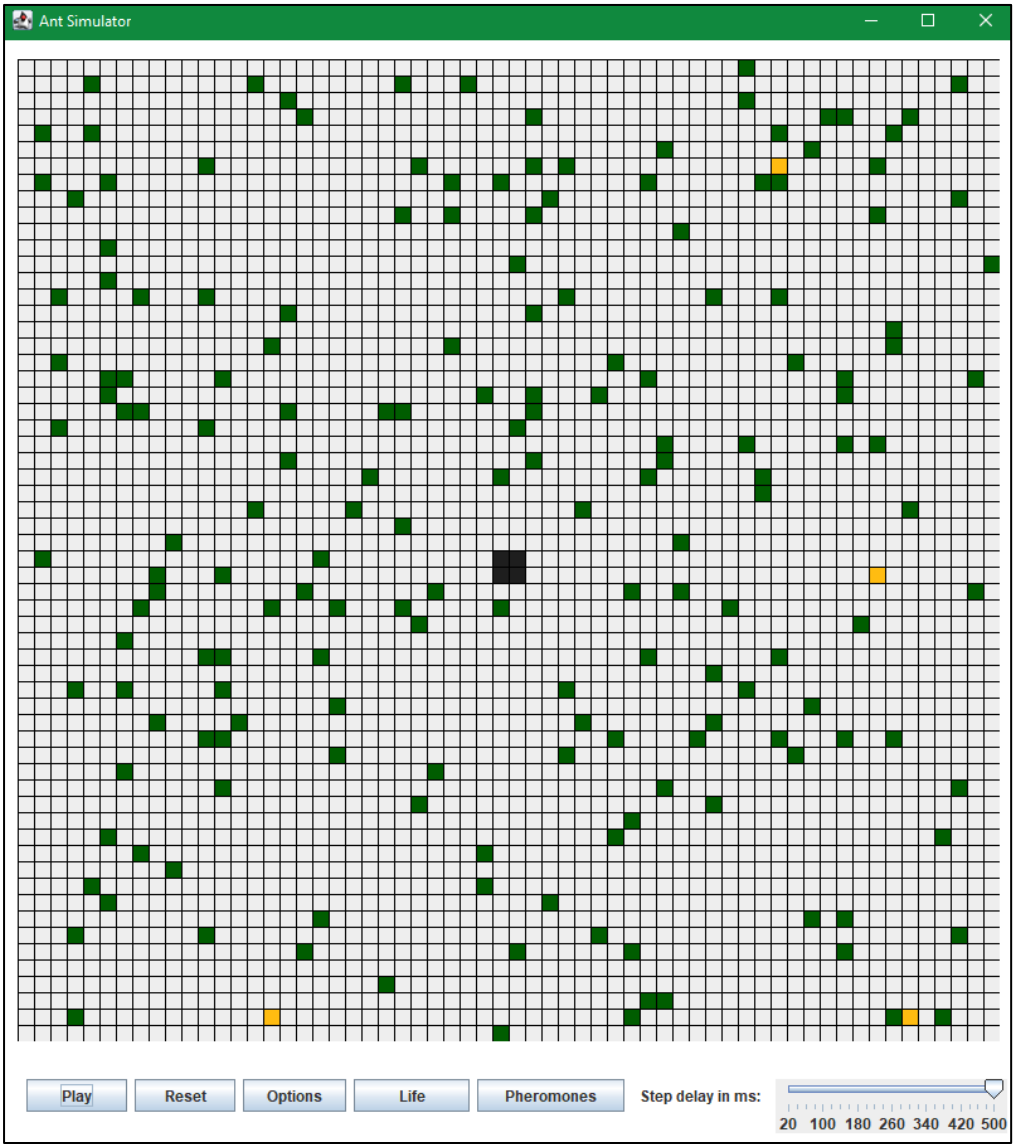
Analisi dei risultati

In seguito all'osservazione e all'analisi dell'esecuzione di molte simulazioni diverse si è potuto subito notare come non ci fosse mai una simulazione esattamente uguale all'altra. Essendo il modello computazionale un CAS, ogni volta che si esegue una simulazione, il modello si comporta in un modo del tutto nuovo e talvolta può presentare risultati unici. Non ci sono due simulazioni totalmente identiche o che producono lo stesso risultato o gli stessi comportamenti emergenti.

Pertanto qui mostriamo una simulazione tipo che ha manifestato numerosi eventi e comportamenti emergenti degni di nota.

Abbiamo lasciato eseguire la simulazione per circa 1000 turni. Ogni turno si aggiorna il frame della GUI e passa un giorno nel modello. Di seguito possiamo visualizzare alcune fasi che commenteremo opportunamente.

Tempo **t = 0**:



Alcune statistiche al tempo t = 0. Le statistiche sono divise in quattro sezioni. Commentiamo brevemente il loro significato:

Day	**0**
Current number of food's sources	4
mean amountLeft	346,5
Current number of trail pheromones	0
mean strength	null
Nest DNA code to give to the next newborn ant	
nTraitsToTransmit	null
changeDirection	null
maxLeaveTrail	null
strengthOfNewTrailPheromone	null
maxStomachCapacity	null
foodToEatEveryDay	null
transferringSpeed	null
maxRoaming	null
Current living ants	180
mean life	99
mean starvingMultiplier	1
mean stomachSum	11,9625
mean roaming	0,994444
mean nTraitsToTransmit	4,577778
mean changeDirection	8,25
mean maxLeaveTrail	22,03889
mean strengthOfNewTrailPheromone	32,95556
mean maxStomachCapacity	12,08333
mean foodToEatEveryDay	0,148537
mean transferringSpeed	0,174897
mean maxRoaming	38,60556

1 – Statistiche relative all'ambiente
Numero del giorno
Numero di fonti di cibo attive
Quanto cibo è presente in ogni fonte in media
Numero di feromoni sulla griglia
Intensità media dei feromoni

2 – Statistiche sul DNA del nido
Gli otto attributi del codice genetico delle formiche memorizzato dal nido, con i loro valori rispettivi

3 – Statistiche generali delle formiche
Numero di formiche vive
Media del valore di vita (100 è il massimo, 0 il minimo)
Indicatore della fame media delle formiche (1 è il minimo)
Somma del cibo nei due stomaci in media
Numero di turni medio spesi esplorando da parte delle formiche

4 – DNA medio delle formiche
Gli otto attributi del codice genetico delle formiche, con i loro valori medi tra tutte le formiche vive

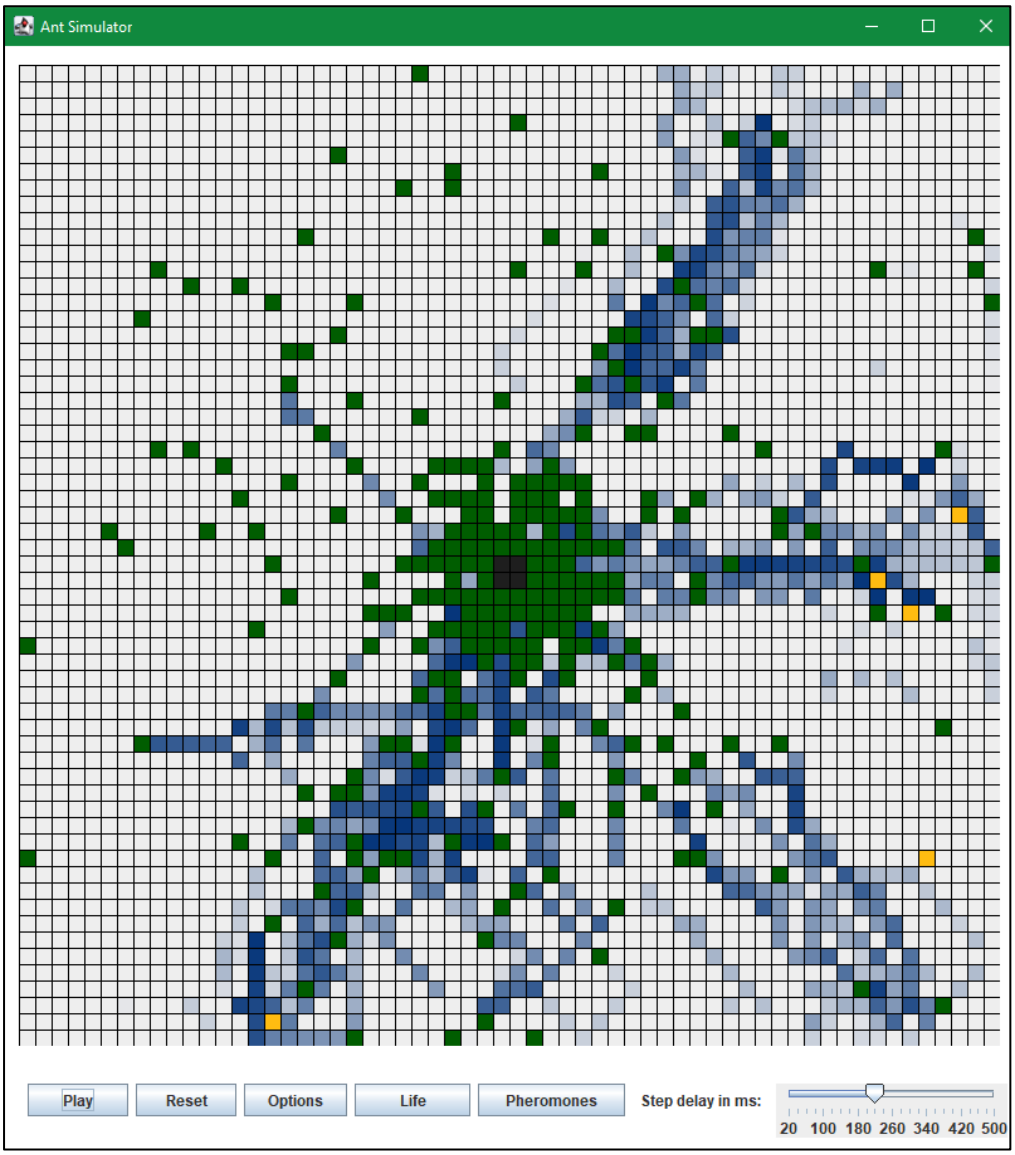
Al tempo 0 le formiche vengono create a caso sulla griglia (l'ambiente del sistema). Ad ogni formica viene dato un insieme di attributi casuale. La media degli otto attributi per ognuno di loro è quasi esattamente a metà tra i valori minimi e massimi (non riportati per brevità), questo vuol dire semplicemente che i generatori di numeri casuali sono veramente a distribuzione uniforme.

Da notare che il nido non possiede ancora nessun valore per gli attributi dato che nessuna formica ha ancora raggiunto il formicaio.

Si nota inoltre che tutte le formiche nascono con cento di vita e possiedono una quantità di cibo nello stomaco in media uguale alla metà tra il massimo ed il minimo che in questo caso è facilmente calcolabile. Il minimo è infatti 0 ed il massimo in media è 24. Infatti "maxStomachCapacity" in media è 12, per due stomaci fa 24.

Il roaming medio uguale a uno è dovuto al fatto che, al momento della raccolta dei dati, un turno è passato e quasi tutte le formiche si sono mosse di una posizione.

Continuiamo la simulazione per un po' di giorni e arriviamo a **t = 80**.



Legenda:

= traccia di feromone

La sfumatura del colore dei feromoni indica la loro forza, più è scuro il colore più è grande l'intensità del feromone.

Statistiche al tempo t = 80.

Day	**80**
Current number of food's sources	5
mean amountLeft	147,3574
Current number of trail pheromones	867
mean strength	24,92503
Nest DNA code to give to the next newborn ant	
nTraitsToTransmit	3
changeDirection	10
maxLeaveTrail	18
strengthOfNewTrailPheromone	47
maxStomachCapacity	12
foodToEatEveryDay	0,253738
transferringSpeed	0,126945
maxRoaming	48
Current living ants	268
mean life	41,98321
mean starvingMultiplier	1,041045
mean stomachSum	10,49882
mean roaming	38,45522
mean nTraitsToTransmit	5,05597
mean changeDirection	8,615672
mean maxLeaveTrail	21,89179
mean strengthOfNewTrailPheromone	32,69403
mean maxStomachCapacity	12,3694
mean foodToEatEveryDay	0,141257
mean transferringSpeed	0,202711
mean maxRoaming	37,6903

Dai dati possiamo subito notare una diminuzione della quantità media del cibo presso le fonti di cibo, segno che le formiche ne stanno raccogliendo molto.

Sulla GUI notiamo già un importante ed affascinante *fenomeno emergente:* La formazione di *strade di feromoni* di varie sfumature di blu dal nido alle fonti di cibo.

Le formiche si sono auto organizzate per creare questi percorsi che facilitano il raggiungimento delle fonti di cibo. Ricordiamo che ogni formica sa solo di lasciare una traccia di feromoni sul percorso che segue a partire da una fonte di cibo trovata. Il punto è che, trovata una fonte, la formica spesso ricalcola la posizione a caso, non va direttamente al nido nella maggioranza dei casi. Solo alcune sono andate direttamente al nido.

Quel che è accaduto è che col tempo, più formiche, passando sulla traccia di feromoni lasciata da altre, trovavano il cibo e contribuivano a potenziare il percorso di feromoni individuato. Nel tempo le tracce che portano più direttamente al nido dalla fonte di cibo si sono intensificate, mentre quelle che portano in altre direzioni per le quali non si arriva da nessuna parte di interessante, difficilmente vengono scelte da più formiche contemporaneamente. Le formiche hanno appreso dei percorsi quasi ottimali con il contributo di tutte, in modo decentralizzato, ovvero a partire da regole bottom-up.

Si è anche osservato come, nei giorni successivi, il percorso a nord del nido che portava ad una fonte di cibo appena esaurita, rapidamente è stato abbandonato dalle

formiche. I feromoni che formavano quell'insieme di strade si è disperso velocemente col minor passaggio di formiche su di esso ed in breve tempo è sparito.

Un'altra cosa che salta all'occhio è la *densità* di formiche intorno al nido. Questo comportamento è normale. Le formiche dopo molto tempo passato in esplorazione tornano al formicaio, alcune depongono uova che generano formiche nuove che vengono posizionate sulla griglia direttamente intorno al nido. In conseguenza ci si deve aspettare che la zona centrale risulti densamente popolata, questo è possibile osservarlo anche in natura. Tuttavia questo comportamento non è costante nel tempo. Ci sono periodi in cui la zona centrale è intensamente popolata come in questo istante, altri in cui la densità è più bassa. Ciò dipende dal numero totale di formiche, dal numero di nuove nate, dalla quantità di fonti di cibo e di strade di feromoni che portano a queste ultime. Infatti una colonia che ha esaurito le fonti di cibo conosciute avrà un tasso di mortalità per fame più alta e quindi un numero minore di formiche nell'ambiente. Ma non c'è da preoccuparsi. L'esplorazione, insita nella natura delle formiche, tende sempre a far sì che almeno una di esse trovi una nuova fonte a cui presto seguiranno le altre e una nuova strada di feromoni si verrà a creare. Ciò è anche garantito dal fatto che l'ambiente è chiuso e limitato, e che ogni giorno, con il 5% di probabilità, nasce in posizione casuale una nuova fonte di cibo sulla griglia.

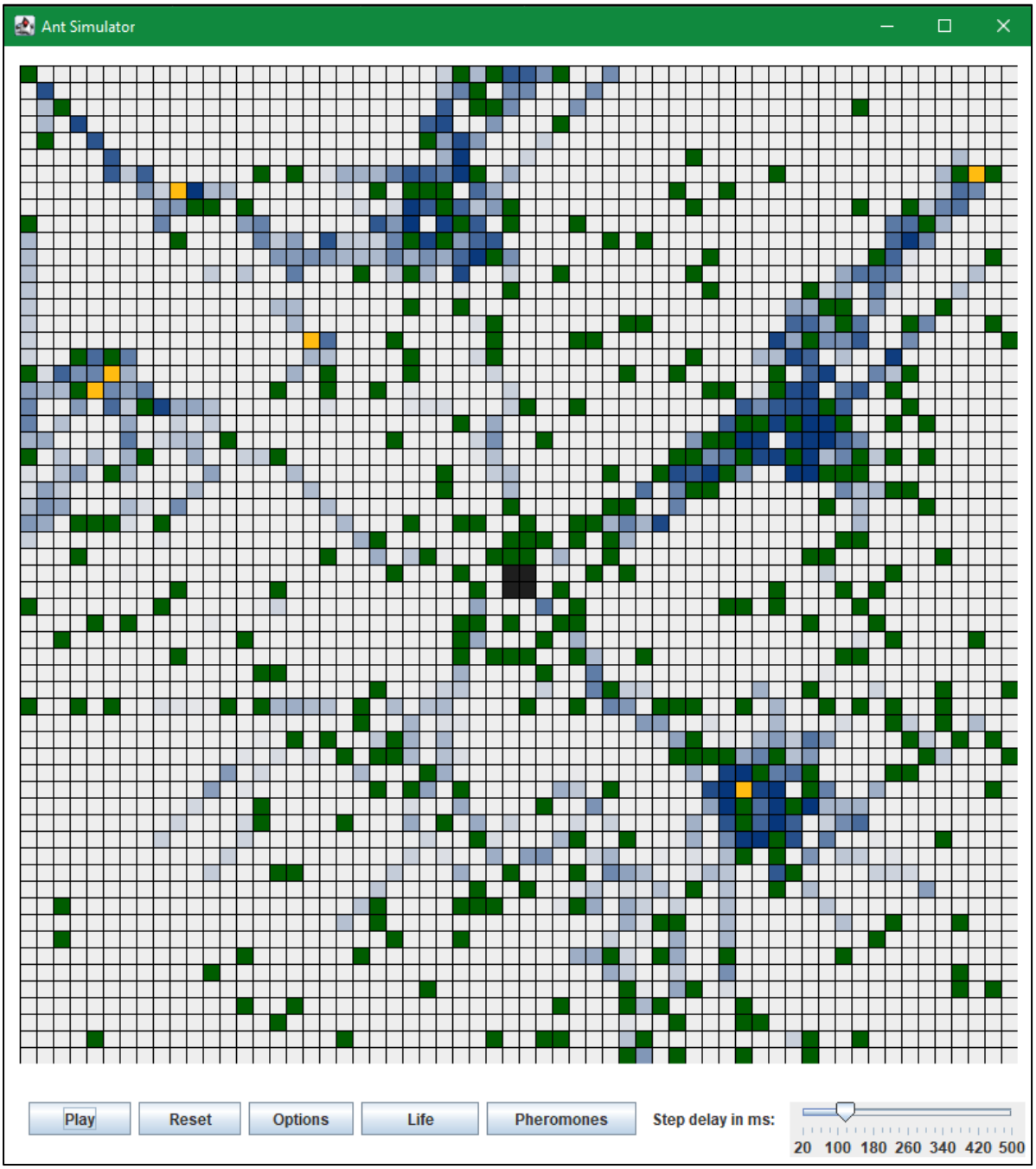
In ultima analisi si può notare come il nido ora possiede effettivamente un valore per ognuno degli otto attributi del DNA. Tutte le formiche che hanno portato cibo al nido, e poi deposto delle uova portanti il loro codice genetico, hanno contribuito a creare tali valori. Infatti ogni formica che depone uova sovrascrive una parte del proprio codice genetico con quello memorizzato nel nido. Ogni nascitura avrà allora nel suo codice genetico una parte che deriva direttamente dal DNA del padre, ed un'altra che è la fusione del DNA di tutte le formiche che sono passate al nido prima del padre.

In conseguenza la media degli otto attributi del DNA nelle formiche sta cambiando.

È cominciato il lungo processo adattivo ed evolutivo.

Mandiamo avanti la simulazione per vedere cosa accade dopo molti giorni.

Visualizziamo in che stato si trova il modello a tempo **t = 1000**.



Una generazione corrisponde a circa 40 giorni di simulazione, corrispondente al numero medio di giorni che impiega una formica per esplorare e tornare al nido del formicaio depositando delle uova. Assumendo 40 giorni per semplicità di calcoli, il millesimo giorno equivale alla *venticinquesima* generazione.

Statistiche al tempo t = 1000.

Day	1000
Current number of food's sources	6
mean amountLeft	170,57
Current number of trail pheromones	655
mean strength	19,58
Nest DNA code to give to the next newborn ant	
nTraitsToTransmit	8
changeDirection	10
maxLeaveTrail	18
strengthOfNewTrailPheromone	17
maxStomachCapacity	15
foodToEatEveryDay	0,025
transferringSpeed	0,39
maxRoaming	35
Current living ants	332
mean life	55,77
mean starvingMultiplier	1
mean stomachSum	13,60
mean roaming	35,25
mean nTraitsToTransmit	8
mean changeDirection	10
mean maxLeaveTrail	17,89
mean strengthOfNewTrailPheromone	16,31
mean maxStomachCapacity	13,14
mean foodToEatEveryDay	0,025
mean transferringSpeed	0,33
mean maxRoaming	36,59

Concludiamo l'analisi dei risultati di questa simulazione con i fenomeni più significativi ed interessanti emersi dal sistema.

Il numero delle formiche vive nel sistema al millesimo giorno, è cresciuto del 100% circa, dall'inizio. Il numero, nel tempo, ha raggiunto un massimo di 600 ed un minimo di 100 unità. Il valore medio oscilla intorno a 400 tra una generazione e l'altra. Sembra che ogni generazione abbia avuto una preferenza sul numero totale di formiche. A lungo andare, comunque, tale valore devia sempre di meno dalla media, segno che il formicaio sta trovando un valore ottimale di formiche nell'ecosistema.

Si è notato come il numero delle formiche vive nel sistema, dipenda, tra le altre cose, dal valore dell'indicatore della fame media del formicaio salvata in "starvingMultiplier". Maggiore è tale valore, minore è il numero di formiche dato che questo indicatore segnala anche l'aspettativa di vita media.

Un aspetto affascinante è che con il tempo tale valore si discosta da uno (il minimo) più raramente. Il sistema si è in qualche modo evoluto per evitare di avere formiche che muoiano di fame.

Passiamo all'analisi più significativa di tutte: sapere come si sono evolute le formiche.

Analizzando i dati sulla media degli otto tratti genetici delle formiche vive nel corso del tempo si possono notare vari fenomeni emergenti.

Primo fra tutti, si nota come la media del numero di tratti passati di padre in figlio è raddoppiata: da 4 a 8. Le formiche che passavano più geni ai figli sono risultate

possedere un genotipo dominante. Questa proprietà non era del tutto programmata ma è emersa dal sistema.

La conseguenza è che, nel tempo, soltanto i figli nati con il codice genetico molto simile al padre, quindi dominante, sopravvivesse e si riproducesse, dato che ne ereditavano tutti gli attributi.

Il secondo aspetto interessante riguarda il notevole cambiamento dei valori del terzo e quarto attributo del DNA delle formiche, rispettivamente "maxLeaveTrail" e "strengthOfNewTrailPheromone". Il primo indica la quantità di giorni consecutivi in cui vengono rilasciati feromoni sul proprio percorso, il secondo l'intensità dei feromoni rilasciati. Contrariamente a quanto ci si aspettava, questi due valori sono scesi di molto. L'ipotesi che abbiamo formulato è che questi valori siano ottimali in rapporto al numero di celle sulla griglia. Ovvero che, data la dimensione dell'ambiente, le formiche abbiano appreso che valori maggiori (per questi due attributi) portavano a strade di feromoni inutilmente lunghe, con conseguente dispendio di tempo ed energie e magari inducendo le formiche a rimanere su percorsi che portavano a fonti di cibo esauritesi nel frattempo. Questo fenomeno emergente è quello che ci affascina maggiormente di più, essendo il più misterioso ed imprevisto.

Il terzo comportamento emergente che abbiamo potuto misurare è stato un leggero incremento della dimensione media degli stomaci delle formiche. Questo tratto è evoluto secondo le aspettative. Era infatti prevedibile che una formica, con una capienza maggiore dello stomaco, avesse maggiori probabilità di sopravvivenza rispetto ad una con uno stomaco di capacità inferiore. Una formica con maggiori chance di sopravvivenza implica direttamente una maggiore probabilità di riprodursi e passare alla prole questo genotipo dominante.

Ultimo comportamento emergente manifestato è la diminuzione della voracità delle formiche. Il valore dell'attributo "foodToEatEveryDay" (letteralmente il cibo da mangiare ogni giorno), dopo 25 generazioni è diminuito costantemente. Anche questo fenomeno era abbastanza prevedibile. Consumare meno cibo ogni giorno garantisce maggiori chance di sopravvivenza e quindi maggior probabilità di riprodursi.

È stato affascinante osservare il modello evolversi, riprodursi, adattarsi al suo ambiente, auto organizzarsi e generare fenomeni emergenti proprio come un formicaio vero. Ne è emersa una intelligenza adattiva che non avevamo programmato e, in alcuni casi, nemmeno previsto. Possiamo quindi affermare con certezza che il modello è conforme alla teoria.

5. Conclusioni

All'inizio di questo lavoro ci siamo posti come obiettivo la comprensione dei sistemi complessi adattivi. Questo obiettivo aveva due principali sotto obiettivi: costruire una teoria per sistemi complessi adattivi e poi provare a costruire un modello computazionale che fosse conforme a tale teoria.

La teoria è risultata essere una fase molto complessa da formulare e strutturare. Il motivo risiede all'origine. Non esiste infatti una disciplina che si occupi propriamente di CAS. Numerosi sono gli scienziati che nella loro vita si sono occupati di questi affascinanti sistemi ma ancora non è emersa, nella comunità scientifica, una disciplina che si occupi di CAS e che fornisca la base teorica sulla quale studiarli e modellarli.

Con grande umiltà, abbiamo provato a formulare questa teoria, a partire dal contributo scientifico di varie fonti. Quest'ultime presentavano spesso l'argomento in modo disorganizzato ma soprattutto non erano sempre concordi su ogni argomento. Infine ci siamo resi conto che avevano una carenza, non inquadravano questi sistemi nella Teoria dei Sistemi.

Per prima cosa abbiamo ritenuto necessario definire con certezza la contestualizzazione dei CAS nella Teoria dei Sistemi. Facendo un breve excursus scientifico, siamo stati in grado di definire alcuni tipi di sistemi dai quali i CAS ereditano delle proprietà. Ci siamo però presto resi conto che non era possibile dare una definizione formale ai sistemi complessi e ancor meno ai sistemi complessi adattivi.

Questa difficoltà non ci ha tuttavia scoraggiato, bensì abbiamo cambiato tattica. Invece di formalizzarli abbiamo provato a descrivere le loro caratteristiche. Abbiamo selezionato solo quelle più significative, senza le quali si sarebbero perse informazioni importanti. Quel che ne è uscito fuori è un importante compendio, completo di tutte le caratteristiche chiave dei CAS. Il primo obiettivo è stato così raggiunto.

Per passare alla modellazione di CAS bisognava tuttavia fare un ulteriore passo. Occorreva prima descrivere quali fossero le qualità e le proprietà da aspettarci e da ricercare in un modello computazionale atto a modellare CAS. Abbiamo selezionato anche qui quelle proprietà che un buon modello computazionale dovrebbe avere. Da questa parte teorica abbiamo concluso che la progettazione di modelli computazionali per CAS richiedeva un paradigma nuovo: la programmazione bottom-up. La filosofia del bottom-up ci ha accompagnato in tutta la teoria e guidato poi nella pratica.

Conclusa la costruzione della teoria siamo potuti passare alla pratica. Per costruire un modello dovevamo scegliere un sistema da modellare. Si è deciso di modellare un formicaio, che è un CAS sociale, perché è un sistema ricco di interessanti proprietà e sul quale è facile reperire fonti di ricerche scientifiche che li studiano.

Abbiamo allora raccontato come funzionano i formicai in natura e poi ci siamo accinti a modellarne uno.

Il modello computazionale che abbiamo costruito, e dal quale abbiamo estratto numerosi dati e raccolto statistiche, ci ha permesso di capire ancor di più alcune dinamiche dei CAS e di sperimentare concretamente il tanto atteso affioramento di pattern e di intelligenza emergente.

Infatti, osservando il modello evolversi sulla GUI e confrontando i dati generati nell'arco delle simulazioni, abbiamo potuto fare esperienza di interessanti fenomeni emersi dal modello che non avevamo programmato. Fenomeni che nemmeno potevamo prevedere potessero affiorare. Abbiamo visto nascere Emergence da un modello computazionale che si fondava su un sistema reale, guidato dalla teoria per CAS che precedentemente avevamo costruito. Abbiamo così veramente raggiunto il cuore dei CAS e potuto confermare che il modello era conforme alla teoria, e la teoria fosse solida e plausibile, essendo direttamente applicabile ad un caso di studio reale.

Le prospettive future per questo lavoro possono essere infinite. Numerosissimi sistemi complessi adattivi possono essere meglio compresi e possono essere costruiti loro modelli grazie ad una base teorica "sound and complete". Tuttavia questo lavoro non può mai considerarsi concluso, non sarebbe corretto altrimenti. La teoria per CAS è essa stessa un sistema complesso, adattivo che evolve nel tempo. Questo aspetto non può che affascinare ed attirare l'attenzione di future ricerche che mirino ad ampliare ulteriormente questa teoria contribuendo alla sua evoluzione, continua, nel tempo.

6. Bibliografia

A proposito di Sistemi Complessi Adattivi:

- Emergence - Steven Johnson
- Complex Adaptive Systems - Miller and Page
- Complex Adaptive Systems - Serena Chan
- Complex Adaptive Systems - John Holland

A proposito di sistemi complessi e pattern in natura:

- Knowledge Management, Organizational Intelligence and Learning, and Complexity - L. Douglas Kiel
- Sezione aurea – Mario Livio
- La Teoria del Tutto – Stephen Hawking

7. Sitografia

Link GitHub dove trovare il software di questo progetto:

- https://github.com/PieMH/Complex-Adaptive-Systems

Link Desmos per visualizzare la funzione probabilistica implementata in "findRandomDirection()":

- https://www.desmos.com/calculator/4x67jrbbux?lang=it

Doppio pendolo:

- https://www.myphysicslab.com/pendulum/double-pendulum-en.html

Chaos Theory:

- https://ocw.mit.edu/courses/physics/8-09-classical-mechanics-iii-fall-2014/lecture-notes/MIT8_09F14_Chapter_7.pdf

A proposito del Physarum polycephalum:

- https://www.pnas.org/content/109/43/17490
- http://social.mbl.edu/how-can-a-slime-mold-solve-a-maze-the-physiology-course-is-finding-out

Formiche e api:

- https://www.britannica.com/animal/ant
- https://www.britannica.com/animal/hymenopteran/Features-of-immature-stages
- https://misfitanimals.com/ants/ant-pheromones

Emergence:

- https://www.youtube.com/watch?v=16W7c0mb-rE

Vita:

- https://www.youtube.com/watch?v=QOCaacO8wus

Intelligenza:

- https://www.youtube.com/watch?v=ck4RGeoHFko

Coscienza:

- https://www.youtube.com/watch?v=H6u0VBqNBQ8