

UNIVERSITÀ DEGLI STUDI DI TORINO



Facoltà di Economia

Corso di Laurea in Economia Aziendale

Tesi di Laurea

Simulazione e Sanità: il 118

Rifatto nel Computer

RELATORE:

prof. Pietro Terna

CORRELATORE:

prof. Sergio Margarita

CANDIDATO:

Carlo Badino

Indice

1	Introduzione	6
2	L'intelligenza artificiale.	10
2.1	L'intelligenza artificiale: tappe fondamentali.	10
2.1.1	L'intelligenza artificiale e il mito.	10
2.1.2	L'intelligenza artificiale e gli automi.	12
2.1.3	Charles Babbage, Ada Byron e il computer.	14
2.1.4	La macchina di Turing e il bambino elettronico.	16
2.1.5	John von Neumann	19
2.1.6	Nasce l'IA.	20
2.1.7	Computer che ascoltano e che parlano.	23
2.1.8	La cibernetica	24
2.1.9	Le reti neurali artificiali.	27
2.1.10	La vita artificiale	30
2.1.11	L'intelligenza distribuita.	33
2.1.12	È possibile l'intelligenza artificiale?	35
2.1.13	L'enigma della stanza cinese.	36
3	La complessità.	39
3.1	Complessità: la scienza del ventunesimo secolo	39

3.1.1	La teoria del Caos	39
3.1.2	La formica di Langton	43
3.1.3	La definizione di complessità.	48
4	La simulazione.	63
4.1	Le simulazioni	64
4.2	Vantaggi delle simulazioni	70
4.3	Problemi delle simulazioni	77
4.4	Le tre grandi separazioni	80
4.5	Simulazione e agenti	88
5	jES.	92
5.1	A cosa serve jES.	92
5.2	Come funziona jES?	97
5.3	L'aspetto What to Do	100
5.4	L'aspetto which is Doing What	106
5.5	OrderDistiller e orderGenerator	110
5.6	Il formalismo When Doing What	111
5.7	La contabilità	113
5.8	I parametri della simulazione	114
6	Il servizio '118'	118
6.1	Il 118.	118
6.2	I dati del modello	124
6.3	Uso di passi computazionali per lanciare ricette.	129
6.4	La pulizia dei dati	145
6.5	Un primo modello ex-ante	149
6.6	Impegni per il futuro.	158

7 Conclusioni	160
A Il Decreto del 27 marzo 1992.	162

Elenco delle figure

3.1	Lo strano attrattore di Lorenz	42
3.2	I primi passi della formica di Langton.	44
3.3	La formica di Langton al passo 11111. La formica sta costruendo la sua ‘autostrada’	45
3.4	Bassa e alta propensione a cambiare comportamento.	58
3.5	Percentuale di crimini all’interno della popolazione in funzione severità del sistema penale	60
3.6	Percentuale di crimini all’interno della popolazione in funzione della deprivazione sociale ed economica.	62
5.1	Lo schema di una ricetta	100
5.2	Un sequential batch	101
5.3	unitData/memoryMatrixes.txt.	102
5.4	Ricette che sfruttano passi computazionali.	103
5.5	Il codice java.	104
5.6	Il codice java bis.	105
5.7	unitData/unitBasicData.txt.	106
5.8	unitData/units.xls.	107
5.9	unitData/endUnitList.txt.	109
6.1	La struttura organizzativa della centrale operativa di Grugliasco	123

6.2	Rami paralleli nelle ricette.	131
6.3	Ricette con passi computazionali.	132
6.4	Il codice c 1999.	133
6.5	Il file unitBasicData.txt.	150
6.6	Il file units.xls.	151
6.7	Lo schema generale di un'unità complessa.	152
6.8	Il file recipes.xls.	153
6.9	Il file orderStartingSequence.xls.	154
6.10	Il modello in condizioni normali.	155
6.11	Il modello sovraccaricato.	155
6.12	Le code si normalizzano con l'introduzione di nuove PVS.	156

Capitolo 1

Introduzione

Lo studio dell'economia neoclassica prevede che gli individui, al momento di prendere una decisione, possano optare:

1. tra un numero limitato di alternative;
2. conoscendo la distribuzione di probabilità degli esiti di ciascuna scelta;
3. in modo da massimizzare il valore atteso della loro funzione di utilità data.

Contrariamente a quanto l'economia neoclassica postula i soggetti economici non possiedono queste caratteristiche.

Le persone, al momento di prendere una decisione, non solo godono di razionalità limitata (sottostanno a limiti di conoscenza e di capacità di analisi), ma si dimostrano inclini a farsi influenzare dalle scelte altrui. Il comportamento umano sembra perciò obbedire a regole complesse che contemplan di volta in volta fenomeni di retroazione positiva e negativa.

Arthur (1994) mette in luce queste caratteristiche grazie ad un breve esempio: el Farol è un bar che può ospitare 100 persone. Se al suo interno sono presenti meno di 60 persone la serata sarà piacevole, per cui altra gente sarà

invogliata a recarsi sul posto (fenomeno di retroazione positiva).

Se all'interno del bar vi sono più di 60 persone, il locale risulterà troppo affollato e la serata sarà meno piacevole, di conseguenza la gente non vorrà rimanere nel posto (fenomeno di retroazione negativa).

Durante un dibattito tenutosi presso il Santa Fe institute nel 1987, il fisico Phil Anderson e l'economista Brian Arthur definirono l'economia alla stregua di un «vetro di spin» (struttura fluida di atomi che interagendo l'uno con l'altro modificano il loro orientamento magnetico).

«Proprio come nei materiali magnetici più comuni, il ferro ad esempio, i componenti fondamentali di un vetro di spin sono atomi di metalli i cui elettroni posseggono un movimento rotatorio intrinseco, o 'spin'. E proprio come nel ferro, ogni atomo genera in virtù di questo spin un minuscolo campo magnetico, che a sua volta costringe l'atomo a esercitare una forza magnetica sullo spin degli atomi vicini. In tal caso però le forze interatomiche agenti in un vetro di spin non determinano un allineamento di tutti gli spin e di conseguenza non risulta un campo magnetico macroscopico, come quello osservabile negli aghi della bussola e nei magneti delle porte dei frigoriferi», Waldrop (1995).

Le forze in un vetro di spin sono orientate in direzioni casuali. Questo disordine su scala atomica indica che un vetro di spin è un miscuglio complesso di retroazioni positive e negative, poiché il suo atomo si sforza di allineare il suo spin in parallelo a quelli vicini e in opposizione a tutti gli altri.

«In generale non esiste alcun modo per realizzare con coerenza un simile proposito; ogni atomo dovrà sempre sopportare una certa frustrazione per doversi allineare contro la propria volontà a quelli vicini. Esistono però molte altre possibilità di disporre gli spin affinché la frustrazione sia abbastanza tollerabile per ogni atomo: uno stato che un fisico descriverebbe 'equilibrio

locale'», Waldrop (1995).

Per Arthur l'economia, come un vetro di spin, presenta un misto di retroazioni positive e negative che genera un numero elevato di stati fondamentali naturali, o punti d'equilibrio.

Ne segue che lo studio dell'economia (e in generale lo studio di ogni fenomeno complesso) devono essere portati avanti seguendo una prospettiva che tenga conto della razionalità limitata delle persone e della loro tendenza a instaurare fenomeni di retroazione positiva e negativa.

Una possibilità per condurre tale studio è data dallo strumento simulativo ad agenti, nato grazie allo sviluppo dell'intelligenza artificiale noto come 'intelligenza distribuita'.

In questo tipo di simulazione gli agenti sono dotati di razionalità limitata e non sono in grado da soli di condizionare in maniera significativa il sistema all'interno del quale agiscono .

Se le azioni dei singoli agenti fanno sì che il sistema nel suo complesso si comporti in maniera 'ordinata', questo risultato emergerà durante il funzionamento della simulazione.

Il mio lavoro si propone di analizzare le basi teoriche che hanno reso possibile l'uso della metodologia ad agenti, partendo dalla nascita dell'intelligenza artificiale e dallo studio della complessità per arrivare alla presentazione dello strumento simulativo.

A questo punto sarà introdotto il programma jES (java Enterprise Simulator) sviluppato dal prof. Pietro Terna a partire dalla biblioteca di funzioni 'Swarm'.

Saranno quindi analizzati i risultati ottenuti dalla simulazione di un'organizzazione reale effettuata grazie a jES (nella fattispecie il servizio di pronto soccorso '118', facente capo alla centrale operativa di Grugliasco, con

competenze per la città e la provincia di Torino).

Capitolo 2

L'intelligenza artificiale.

2.1 L'intelligenza artificiale: tappe fondamentali.

Può una macchina pensare? Può un congegno elettronico essere sofisticato a tal punto da 'prendere vita'?

Lo studio della realtà artificiale si è spesso proposto interrogativi di questo tipo. Guardiamo da vicino le tappe che hanno portato alla nascita di questa branca della scienza.

2.1.1 L'intelligenza artificiale e il mito.

L'idea di riuscire a dar vita alla materia inerte ha affascinato l'uomo fin dall'antichità. Nell'antica Grecia nasce il mito di Pigmalione che si innamora perdutamente di una statua raffigurante una donna bellissima da lui stesso scolpita. Gli dei, mossi a compassione, danno vita alla statua.

Si narra che a fine Duecento due frati francescani, Ruggiero Bacon e Bungey, per difendere l'Inghilterra dagli invasori progettassero di costruire un'immen-

sa muraglia d'ottone che corresse lungo i confini del regno. Per realizzare l'opera i due frati progettano prima di tutto una testa meccanica che calcoli e spieghi come costruire il muro. Una volta realizzata la testa non parlerà se non grazie all'intercessione di uno spirito che ammonirà Bacone e Bungey avvertendoli di aspettare il momento in cui l'artefatto aprirà bocca.

Miti analoghi sono comuni in quasi tutti i paesi europei. Costruttori di automi sarebbero stati tra gli altri papa Silvestro II, sant'Alberto Magno e Leonardo da Vinci, mentre Paracelso avrebbe creato un essere umano vivente per via alchemica.

Il mito più famoso è sicuramente quello del Golem d'argilla, mostro della mitologia ebraica ricomparso nella Praga del Cinquecento. Judah Loew ben Bezalel, rabbino capo della città, avrebbe fatto rivivere il mostro per difendere gli ebrei del ghetto di Praga dai pogrom scatenati dai cristiani.

Raccontano Castelfranchi e Stock (2000) che il rabbino e due assistenti scesero notte tempo sul greto della Moldava e che con l'argilla del fiume impastarono una figura umanoide. Il primo assistente compì un giro attorno alla statua per sette volte, da sinistra verso destra, mentre Loew pronunciava il nome segreto dell'Eterno. Golem si accese dello splendore di cento fiamme. Il secondo assistente ripeté il procedimento, compiendo però i giri da destra verso sinistra. Ancora una volta il rabbino capo pronunciò le parole segrete: Golem si spense, ma sulla sua testa crebbero capelli e unghie sulle sue mani. Infine lo stesso Loew compì sette giri intorno alla creatura di fango e tracciò sulla fronte di Golem il nome EMET (verità).

Nella storia Golem, da difensore degli ebrei, diventò presto un mostro incontrollabile. Per ucciderlo si dovette cancellare l'alef (la prima lettera di EMET) dalla sua fronte, perché restasse soltanto MET (morte).

I toni cupi di questo mito e il finale in cui la creatura si rivolta contro il

suo creatore suonano ad ammonizione: l'uomo che abbandona il legame con il divino è morto nell'anima quanto un cumulo di materia inerte. Le radici della vita sono appannaggio esclusivo di Dio.

Più di recente una rivisitazione del mito del golem è stata fornita dallo scrittore argentino José Luis Borges ne *Las ruinas circulares* (1977), pubblicata anche come sotto il nome di *El Golem*. Nel racconto di Borges un uomo dedica la sua esistenza al compito pensare un essere umano e di dare vita al proprio pensiero. Il protagonista del racconto riuscirà nell'impresa grazie all'intervento del dio Fuoco che chiederà in cambio che la creatura si dedichi a perpetrare il suo culto.

Due saranno i doni del Fuoco al nuovo nato: Non avrà ricordi che tradiscano la sua natura artificiale e nessuno all'infuori di suo padre e del Fuoco stesso lo potrà distinguere da un essere umano comune. Alla fine del racconto, il padre della creatura si trova intrappolato tra le fiamme di un incendio, ma queste non lo bruciano. Capisce così il protagonista di essere venuto al mondo alla stessa maniera di suo figlio e di essere anch'esso una creatura artificiale.

2.1.2 L'intelligenza artificiale e gli automi.

L'uomo non si è accontentato di dare vita al fango solo nella finzione del racconto. Moltissimi ingegneri lungo il corso della storia hanno cercato di costruire apparati meccanici dalle sembianze umane o animali che riuscissero a muoversi da sole. uno dei primi esempi di macchine meccaniche che la storia ricordi si ebbe durante l'assedio romano a Cartagine. La città sembrava inespugnabile grazie alle macchine anti assedio progettate da Archimede, in grado di afferrare le navi dall'acqua e di lanciarle sulle rocce o di scagliare massi a grande distanza.

Durante il medioevo si vide un rifiorire della tecnologia meccanica: ingranag-

gi e ruote dentate furono perfezionati per costruire i mulini a vento e le ruote idrauliche furono utilizzate per macinare il grano. Il diffondersi dell'orologio portò la tecnologia degli ingranaggi a livelli mai raggiunti prima. Nel 1540 l'artigiano cremonese Gianello della Torre costruì per l'imperatore Carlo V una giovinetta meccanica che suonava il liuto, Castelfranchi Stock (2000). La giovinetta sapeva camminare in linea retta, si guardava intorno e suonava pizzicando davvero le corde.

La moda di creare automi non era esclusivamente europea, anche in Giappone, già dall'840 d.C. comparvero le prime marionette meccaniche e in India nel 1792 il sultano Tipu Sahib fece costruire una tigre meccanica che divorava il pupazzo di un soldato inglese.

I due automi più famosi sono comunque lo scrivano automatico di Pierre Jaquet-Droz (1774) e l'anitra meccanica di Jacques de Vaucanson (1738). Il primo era parzialmente programmabile ed era in grado di immergere il pennino e scrivere una quarantina di caratteri decisi dall'utente. La seconda sembra fosse in grado di nuotare, starnazzare, bere acqua e mangiare granaglie.

Tutti questi esseri artificiali non avevano nulla a che vedere con l'intelligenza: erano marionette più o meno sofisticate.

Lo sviluppo dell'orologeria permise di inseguire un altro sogno antico: quello di meccanizzare e automatizzare il calcolo matematico e il ragionamento logico, Castelfranchi e Stock (2000). Nel 1624 Wilhelm Schickhard costruisce a Heidelberg il primo orologio calcolatore capace di effettuare le quattro operazioni. Nel 1642 Blaise Pascal costruisce a Parigi una macchina calcolatrice numerica per regalarla al padre esattore, affaticato dal calcolo della ripartizione delle tasse in Normandia.

Nel 1673 Gottfried Leibniz costruisce una sua versione perfezionata della

macchina calcolatrice. Il congegno costruito dal grande filosofo sapeva eseguire i più semplici fra i processi di ragionamento, quelli dell'aritmetica elementare, ma il progetto muoveva da un'idea di portata più grande: Leibniz aveva l'obbiettivo di comprendere e riprodurre i meccanismi del ragionamento astratto.

Scrivono Castelfranchi e Stock (2000) «Come diplomatico si era trovato di fronte ad esempi concreti e drammatici di come l'incomprensione o il disaccordo fra persone potessero portare a conseguenze disastrose». Da qui la volontà di risolvere i conflitti individuando una tecnica imparziale, automatica di scoprire in ogni disputa concettuale chi avesse ragione. Per far ciò occorreva individuare il linguaggio ideale: quello della logica, capace di rappresentare simbolicamente e in maniera inconfutabile ogni situazione.

Completato tale linguaggio universale, disse Leibniz, le discussioni filosofiche non sarebbero più necessarie di quanto lo siano quelle tra due contabili. Basterebbe infatti che essi [i filosofi] prendessero in mano le loro penne, si mettessero ai loro tavoli e si dicessero a vicenda (con un amico come testimone se lo desiderano): calcoliamo.

2.1.3 Charles Babbage, Ada Byron e il computer.

Durante una serata culturale a casa di Mary Sommerville, una sera di novembre del 1834, Charles Babbage, scienziato inglese, presenta una nuova invenzione. Una macchina per calcolare differente da quelle fino ad allora prodotte. Una macchina universale che fosse in grado di eseguire qualsiasi calcolo numerico e logico, dai logaritmi alle orbite dei pianeti.

Presente tra gli altri in casa Sommerville era Ada Byron, la figlia del celebre poeta. Quella sera sarebbe nato un sodalizio tra Babbage e Ada Byron. Lui

sarebbe passato alla storia come l'inventore del computer, lei come prima programmatrice della storia.

Nel 1819 Babbage ha in mente il progetto di una macchina addizionatrice basata sul metodo delle differenze finite (la Difference Engine). Babbage non riuscì a portare a termine la macchina, che fu completata nel 1853 da George Scheutz, con grande gioia del suo ideatore. Nonostante il precedente fallimento Charles Babbage nel 1833 ha un'idea ancora più ambiziosa: una macchina programmabile capace di calcolare qualunque cosa e di compiere operazioni logiche.

L'architettura logica e le funzioni svolte dagli ingranaggi della macchina analitica sono simili a quelle dei moderni computer elettronici. Babbage descrive cinque componenti dell'apparecchio:

- un magazzino: il luogo dove vengono conservati i dati numerici su cui operare;
- la macina: il luogo in cui i dati vengono manipolati e cambiati;
- il controllo: la parte del computer che gestisce il programma;
- l'input e l'output: apparati con i quali il calcolatore riceve dati e comunica con l'esterno, Castelfranchi e Stock (2000).

Ada Byron si rese conto per prima che la novità della macchina analitica rispetto alle calcolatrici meccaniche consisteva nella possibilità di manipolare simboli generici e non soltanto numeri. La macchina poteva eseguire operazioni simboliche astratte. Ada inventò i primi programmi e pose così le basi della tecnica di programmazione.

A lei dobbiamo l'invenzione dell'istruzione condizionale e dell'iterazione.

Nonostante questo non pensò mai che le capacità logiche e analitiche di una

macchina avrebbero potuto portare ad una vera Intelligenza Artificiale, al contrario preconizzò una delle obiezioni all'idea di macchina pensante: una macchina non potrà mai possedere un'intelligenza perché sa solo eseguire un programma stabilito.

2.1.4 La macchina di Turing e il bambino elettronico.

Nel 1900 a Parigi, David Hilbert lanciò una sfida ai matematici di tutto il mondo. Individuò venti tre problemi insoluti da anni o da secoli e da lui considerati di importanza cruciale.

Nel 1928, Hilbert rilanciò tre problemi scelti tra quelli che riguardavano le basi logiche di tutta la matematica: quello della coerenza (come possiamo essere sicuri che la matematica non nasconda contraddizioni tra i suoi assiomi?), quello della completezza (siamo certi che in matematica non esistano enunciati veri, ma non dimostrabili?) e quello della decidibilità (esiste sempre una maniera rigorosa per stabilire se un certo enunciato matematico sia vero o falso?), quest'ultimo noto anche come Entscheidungsproblem.

Un caso famoso di enunciato matematico di cui sia incerta la verità è costituito dalla congettura di Goldbach secondo la quale tutti i numeri interi pari, maggiori di due, sono la somma di due numeri primi.

Nel 1931 il logico austriaco Kurt Gödel aveva chiarito i primi due quesiti. Nel 1936 Alan Turing risolse il terzo grazie ad un'idea che gli era venuta guardando le nuvole sdraiato su un prato di Grantchester.

L'idea è quella di un semplice mezzo meccanico capace di eseguire operazioni logiche. Turing la chiama macchina di computazione logica e la immagina composta di un nastro di carta scorrevole diviso in caselle, un pennino per scriverci sopra e un apparecchio che può decidere di far scorrere il nastro avanti o indietro, di farci un segno o di cancellarne uno già presente.

Scrivono Castelfranchi e Stock (2000) «la macchina scrive o modifica i segni sulla carta obbedendo a certe regole logiche imposte dal costruttore. Per esempio, la macchina di Turing potrebbe avere il semplice scopo di scrivere una X su ogni casella del nastro, oppure di cancellare ogni X che incontra [...]. Ma Turing andò oltre: capì che non era necessario immaginare innumerevoli tipi di macchine diverse [...]. Ne bastava una sola, a patto che potesse ‘imparare’, ad esempio tramite i dati scritti sul nastro di carta, le regole da seguire caso per caso». Oggi chiamiamo questo concetto macchina di Turing universale e costituisce in nuce l’idea moderna di computer: una macchina capace di imitare qualsiasi macchina logica basandosi su dei programmi.

Turing capì che esistevano dei problemi che la macchina non avrebbe saputo risolvere e dimostrò che esisteva un ‘problema dell’arresto’: «può accadere quando scriviamo un programma per risolvere un problema che sia impossibile sapere se il calcolatore fornirà una risposta o continuerà a calcolare all’infinito». Di conseguenza se una macchina poteva rappresentare il linguaggio della matematica e se esistevano per quella macchina problemi indecidibili, allora quei problemi erano indecidibili per tutti. La risposta all’Entscheidungsproblem è negativa: non è possibile stabilire in maniera meccanica se un teorema sia dimostrabile.

La macchina di Turing oltre a dare una risposta ad uno dei problemi più interessanti della matematica mostra con chiarezza che oltre al calcolo numerico anche i ragionamenti logici e i pensieri astratti possono essere affrontati da un sistema meccanico.

L’avvento del calcolatore scatenò in Alan Turing pensieri profondi, il ricercatore si chiese quanto fosse possibile per una macchina calcolatrice simulare le attività umane. Si convinse che per creare una mente elettronica occorreva un computer capace di imparare e di modificarsi in funzione dell’ambiente

esterno. Turing capiva benissimo che l'intelligenza è fatta anche di creatività e che la mente non esiste se non connessa ad un corpo. Pensò allora che i computer del futuro dovessero essere in grado di interagire con il mondo esterno, di poter «vagare in giro per le campagne» e di conoscere il mondo. Conscio del fatto che ci sarebbero stati sempre campi dell'attività umana dai quali i computer sarebbero stati esclusi, elencò una serie di settori, come il gioco degli scacchi, dove le macchine avrebbero potuto fare pratica.

Nel 1950 Turing scrisse un articolo che sarebbe divenuto il più celebre della disciplina dell'Intelligenza Artificiale: immaginò una macchina capace di riprogrammarsi da sé, Castelfranchi e Stock (2000), di osservare il mondo esterno e di mutare il proprio comportamento di conseguenza.

Immaginò il bambino elettronico e propose, in alternativa alla domanda 'le macchine possono pensare?', di esprimere il problema nei termini di un gioco: il gioco dell'imitazione.

Nel gioco dell'imitazione una persona è chiusa all'interno di una stanza e può comunicare con l'esterno solo per via di una tastiera e di uno schermo. Allo stesso modo in altre due stanze comunicano altre due persone, una di sesso maschile e l'altra di sesso femminile.

La prima persona può formulare delle domande alle quali le altre due devono rispondere. Nel formulare la risposta esse hanno però diritto di mentire. Scopo del gioco è per la prima persona indovinare quale dei due interlocutori sia maschio e quale sia femmina. Per le altre due celare il proprio sesso.

Turing propone una variante: uno dei due interlocutori sarà un computer e la prima persona dovrà indovinare la natura, umana o artificiale, dei suoi interlocutori. La domanda che ci si pone è la seguente: cosa cambierebbe tra una prova e l'altra? La prima persona sbaglierebbe l'identificazione la stessa percentuale di volte?

Continua Turing, «non sapendo ancora definire cosa siano il pensiero, la conoscenza e l'intelligenza, se una macchina universale riuscisse ad ingannare un essere umano celando la sua natura artificiale, se riuscisse a reagire in tutto e per tutto come una persona, non saremmo costretti a concludere che essa 'pensa'?»

2.1.5 John von Neumann

John von Neumann, matematico ungherese, ebbe un ruolo fondamentale nello sviluppo del calcolatore elettronico.

Neumann, posto di fronte a calcoli complicatissimi nell'ambito del progetto Manhattan, si rese immediatamente conto delle possibilità che avrebbe offerto la potenza di calcolo del computer. Con Presper Eckert e John Mauchly progettò nel 1945 il primo computer moderno: EDVAC (Electronic Discrete Variables Automatic Computer). Edvac vide la luce nel 1952 e fu la prima macchina digitale programmabile tramite un software.

Prima ancora che EDVAC nascesse, a Cambridge fu costruito EDSAC (Electronic Delay Storage Automatic Calculator), la prima macchina a contenere un linguaggio assembler. Nel 1951 sarebbe nato Univac I, il primo computer commerciale, costruito sul progetto di EDVAC, ma messo in commercio un anno prima della sua ultimazione.

«Von Neumann intuì che il calcolatore poteva rappresentare il primo prototipo di una nuova categoria di strumenti: un'estensione della mente umana», Castelfranchi e Stock (2000). Ipotizzò che esistesse un parallelismo tra il computer e il cervello umano. Individuò alcune analogie tanto nella struttura che nelle funzioni dei due oggetti e pensò di sfruttarle per formulare teorie logico-formali capaci di spiegare tanto la mente quanto il computer.

2.1.6 Nasce l'IA.

Secondo Castelfranchi e Stock (2000) l'intelligenza artificiale nasce ufficialmente nell'agosto del 1956 a Dartmouth College, Hanover, New Hampshire. Il matematico John McCarthy organizza un seminario estivo incentrato sull'ipotesi che 'ogni aspetto dell'apprendimento o ogni altra caratteristica dell'intelligenza possono essere, in linea di principio, descritti in modo così preciso da essere simulati mediante una macchina'.

Il seminario, chiamato 'Summer Research Project on Artificial Intelligence' vede la partecipazione di ricercatori provenienti dalle più diverse discipline, tra gli altri Claude Shannon e Marvin Minsky. Lo scopo degli studiosi presenti è di studiare come costruire 'macchine che usino il linguaggio, formino astrazioni e concetti, risolvano classi di problemi ora riservate agli umani, migliorino se stesse'.

Durante il seminario si verifica un evento importante. Cliff Shaw, Allen Newell e Herbert Simon si presentano con un programma 'intelligente' capace di scoprire dimostrazioni di teoremi matematici: 'Logic Theorist'.

Il programma era in grado di dimostrare i più astratti tra i teoremi contenuti tra i *Principia Mathematicae* di Alfred North Whitehead e Bertrand Russell. L'idea di base è semplice: esiste un modo banale di dimostrare tutti i teoremi possibili: basta prendere uno a uno i postulati di partenza e applicare ad essi tutte le possibili combinazioni di regole di inferenza. In questo modo si ottiene una serie di teoremi. A questi teoremi saranno applicate nuovamente le regole di inferenza.

Il problema di questo procedimento è che si ottiene un numero impressionante di teoremi banali, in mezzo ai quali sarà impossibile riconoscere quelli interessanti.

Diventava essenziale che la macchina fosse in grado di indagare solo nelle

direzioni più promettenti. Per far ciò Logic Theorist doveva avere a disposizione un metodo euristico per operare le scelte.

La strategia che si applicò fu la seguente: Logic Theorist non partiva dai postulati, ma dalla forma della dimostrazione da individuare e cercava di raggiungere a ritroso i postulati iniziali.

Un anno dopo Newell e Simon produssero un programma ancora più ambizioso: The General Problem Solver (Gps). Gps era in grado di affrontare una gran varietà di problemi logico-matematici avvalendosi dell' 'analisi mezzo-fine'. Il calcolatore, secondo questa logica, conosce gli stati iniziale e finale di un problema e ricerca le operazioni possibili che permettano di ridurre via via la distanza tra questi due stati.

Un secondo importante prodotto dell'IA furono i sistemi esperti: programmi dotati di scarsa intelligenza, ma che conoscevano alla perfezione alcuni settori dello scibile umano.

I sistemi esperti furono un grande successo tecnico e commerciale. Essi erano costituiti da almeno due parti fondamentali: 'una base di conoscenza' e un 'motore inferenziale'. La base di conoscenze è la parte che deve cercare di raccogliere e rappresentare in maniera simbolica tutte le conoscenze che un esperto umano usa per risolvere un problema.

Alla base della capacità di ragionare, di 'inferire' è l'altra componente fondamentale, il motore inferenziale, che è costituito da un insieme di regole logiche che permettono di dedurre un fatto nuovo a partire dalle informazioni disponibili.

I sistemi esperti si rivelarono alla prova dei fatti incredibilmente efficaci. Mycin, esperto in diagnosi sulla meningite e sulle infezioni del sangue, fu sottoposto ad un esame nel 1979: «analizzò i casi di dieci pazienti reali insieme ad altri nove medici. Le terapie suggerite dal programma e dai medici

furono poi sottoposte alla valutazione di otto esperti, i quali non sapevano chi fossero gli autori delle diagnosi e delle terapie. Il punteggio totale poteva andare da 0 a 80. La classifica che ne scaturì mostrò che Mycin si comportava come i migliori medici del settore (anzi un po' meglio seppure in maniera non significativa)», Castelfranchi e Stock (2000).

Per molti aspetti i sistemi esperti hanno rappresentato un successo dell'IA, ma possiamo considerarli veramente intelligenti?

I sistemi esperti sono rigidi, non hanno creatività e non hanno idea del mondo esterno.

Un sistema esperto che veniva utilizzato presso le concessionarie Ford per decidere se concedere finanziamenti agevolati ai clienti concesse un prestito favoloso ad un ragazzo di vent'anni che sosteneva di lavorare da dieci anni.

Per superare questi limiti Douglas Lenat decise che l'unica macchina che sarebbe stata in grado di dimostrare senso comune sarebbe stata quella che avesse avuto una conoscenza completa a proposito del mondo umano. Nacque così, nel 1984, Cyc.

«Secondo Lenat il fattore cruciale che distingue il pensiero di noi umani da quello dei computer è il fatto che questi ultimi non sanno quasi nulla del mondo che li circonda, mentre ogni persona ha un bagaglio di conoscenze e di senso comune che gli permette di interpretare i fatti, di imparare e comunicare. La differenza fra un computer pensante del futuro e i Pc di oggi non starebbe tanto in una potenza di calcolo fantascientifica o in un software magico in attesa di essere scritto, ma semplicemente in una immensa base di conoscenze sui fatti e sulle regole che governano il nostro mondo», Castelfranchi e Stock (2000).

Cyc conosce centinaia di migliaia di fatti apparentemente banali che riguardano la vita di tutti i giorni, grazie ai quali è già in grado di cogliere la

differenza tra frasi strutturate in maniera molto simile e di individuare soggetti con diverse caratteristiche cercandoli in un repertorio fotografico. Secondo Lenat, man mano che il database delle conoscenze di Cyc cresce, questo diventa sempre più intelligente e ipotizza che per il 2025 potrà essere un essere pensante completo.

2.1.7 Computer che ascoltano e che parlano.

La comunicazione linguistica è una delle caratteristiche da sempre considerate fondanti dell'intelligenza umana.

Eliza, un famoso programma di conversazione fu scritto fra il 1964 e il 1966 da Joseph Weizenbaum, poteva interpretare vari ruoli a seconda dell'archivio di frasi che le veniva associato. Il ruolo più famoso che Eliza poteva interpretare era quello di 'Doctor', uno psicanalista di scuola rogeriana (scuola in cui il terapeuta non analizza ciò che il paziente dice, ma si limita ad incoraggiare quest'ultimo a parlare).

Eliza era un software 'stupido', il meccanismo in base al quale funzionava si basava sul riconoscimento di parole chiave o di schemi linguistici. Nonostante questo negli anni '60 il 'dottore' conobbe una straordinaria celebrità.

Kenneth Colby si ispirò ad Eliza per creare un paziente virtuale che soffriva di disturbi di tipo paranoide, Parry.

In Eliza, come in Parry, non c'è traccia di intelligenza artificiale. Ciononostante il comportamento di Parry, sottoposto ad analisi da parte di diversi psichiatri, risultò effettivamente compatibile con un quadro clinico paranoico.

Douglas Hofstadter (1984) racconta dell'incontro tra Eliza e Parry, che si svolse come segue, suscitando l'ilarità dei ricercatori:

PARRY: Non capisco le tue ragioni...

DOTTORE: Perché non capisci le mie ragioni?

P: Non dovrei forse?

D: Perché dici ciò proprio ora?

P: Semplicemente perché le cose stanno così.

D: Supponi che invece non stiano così.

P: E allora?

D: Non sono sicuro di riuscire a capirti del tutto.

P: Pizzichi e suoni troppo sulle stesse domande...

D: A cosa ti fa pensare questa similitudine?

P: Non afferro.

D: E vorresti afferrare?

P: È riguardo il tempo in cui mostravi qualche interesse verso i miei sentimenti.

D: Eh? Ho mostrato qualche interesse verso i tuoi sentimenti?

2.1.8 La cibernetica

Norbert Wiener è considerato il padre della cibernetica, «la scienza del controllo e della comunicazione nell'animale e nella macchina», di cui egli coniò il nome.

La cibernetica nacque grazie agli studi compiuti da Wiener e da altri ricercatori a proposito del meccanismo di retroazione (feedback), che anima parte del controllo dei muscoli di alcuni animali.

Il meccanismo di feedback si comporta come segue: una parte del risultato di un'azione che un sistema compie viene ritrasmessa al sistema stesso, in modo che esso possa calibrare l'azione successiva in funzione di quanto accade al momento nell'ambiente.

Grazie agli studi di Weiner nacquero una serie impressionante di creature

cibernetiche basate sulla percezione del mondo esterno e sul principio del feedback.

Nel 1948 l'inglese Walter Gray Walter costruì Elsie e Elmer, due tartarughe elettroniche attratte dalle sorgenti luminose. In seguito costruì una terza tartaruga, Cora, che lo studioso era riuscito ad addestrare perché accorresse al richiamo di un fischiotto (l'addestramento aveva avuto luogo producendo un suono con il fischiotto ogni qual volta Cora si trovava di fronte ad una sorgente luminosa).

Da questi esempi si evince la differenza che esiste tra la cibernetica e l'intelligenza artificiale, la prima si ripropone di ricostruire i riflessi nervosi degli esseri viventi. Questa filosofia è definita 'bottom-up' (dal basso verso l'alto) e cerca di arrivare alla creazione di esseri artificiali intelligenti partendo dalla ricostruzione del loro sistema nervoso.

L'intelligenza artificiale, al contrario, cerca per prima cosa di ricreare i pensieri astratti, la facoltà di ragionare. Questa seconda filosofia è chiamata top-down (dall'alto verso il basso).

Sfruttando la cibernetica, Rodney Brooks, verso la metà degli anni '80, ebbe l'idea di costruire automi privi di un cervello elettronico, ma dotati di molti riflessi nervosi interconnessi in maniera complessa in modo da far emergere l'intelligenza. I primi prodotti ricavati da questa idea furono Genghis e Attila, due insetti-robot capaci di imparare a camminare coordinando le numerose zampe di cui erano dotati e di esplorare il territorio in cui vivevano. Brooks tuttavia si spinse oltre, convinto che l'intelligenza fosse una proprietà emergente dei sistemi complessi, decise nel 1992 di costruire un bambino elettronico basato su quella che lui chiamava un'architettura di sussunzione (per la quale un robot deve reagire agli stimoli provenienti dal mondo esterno secondo serie di livelli di relazioni fra le percezioni e le azioni dotate di strati

via via più complessi). Nacque così Cog.

«Cog non ha tutta la coscienza che possiedono un bambino, un cane o un uccello, ma ne ha almeno parte», Schechter (1999). Brooks aggiunge che Cog è un vero bambino elettronico. Dotato di un braccio solo e senza gambe, osserva e tocca il mondo come farebbe un bambino vero, cercando di afferrare la mano che gli si tende senza essere programmato per farlo.

Non ha un modello predefinito di cosa sia il mondo e di come comportarsi.

«Il suo cervello non è un organo centralizzato, ma è fatto di oltre un centinaio di chip di personal computer sparsi per il corpo e addetti a gestire i dati delle telecamere e dei microfoni [di cui è dotato], a controllare i movimenti della testa e del braccio, a produrre i suoni che [...] emette.

Il software è assai ridotto, e serve a gestire gli schemi di reazione fra stimolo e risposta: niente regole di inferenza, niente logica formale, niente analisi del linguaggio.

Sarà Cog stesso che dovrà plasmare tali reazioni per tentativi ed errori, come un bambino, sino a imparare a coordinare vista e movimento, a capire le frasi che ascolta, a rispondere con la propria voce», Castelfranchi e Stock (2000)

.

Kimset, il fratello minore di Cog, ha solo la testa, ma è dotato di sopracciglia e di altri attributi facciali che lo rendono stranamente espressivo.

2.1.9 Le reti neurali artificiali.

Tra gli argomenti di studio sull'intelligenza artificiale che si avvicinano al problema secondo una filosofia bottom-up, grande importanza rivestono le reti neurali artificiali.

Le reti neurali contenute nel cervello umano sono costituite da neuroni, ogni neurone appartenente alla corteccia cerebrale è una cellula in grado di ricevere ed inviare messaggi a molti suoi simili.

Il messaggio in uscita, sotto forma di impulso elettrico, corre lungo un ramo chiamato 'assone', i messaggi in entrata passano attraverso i 'dendriti', filamenti sottilissimi che connettono il neurone agli assoni dei neuroni circostanti.

I collegamenti tra dendrite e assone si chiamano 'sinapsi'. Le sinapsi possono essere classificate come eccitatorie o inibitorie a seconda che amplifichino o deprimano i segnali che passano dagli assoni ai dendriti.

Il neurone funziona in maniera semplice, se i segnali in entrata superano una certa soglia, esso genera a sua volta un segnale in uscita.

Nel 1949 Donald Hebb ipotizzò che la memoria e il pensiero umano non fossero direttamente connessi con i neuroni o con gli impulsi che questi si scambiavano, ma che dipendessero dalla struttura della rete formata dallo scambio di impulsi tra cellula e cellula.

L'idea si rivelò sostanzialmente esatta. Due ricercatori, McCulloch e Pitts, si occuparono di tradurla in forma logico-matematica. Nel 1958 Frank Rosenblatt costruì le prime reti neurali artificiali, macchine in grado di imparare, di possedere concetti in maniera non simbolica. Erano strutture basate su una rete di oggetti semplici invece che su algoritmi e concetti logici.

Terna (1994) definisce una rete neurale artificiale, in estrema sintesi, una funzione che «produce un vettore di output da un vettore di input, sulla base

di un insieme di parametri (detti pesi) la cui determinazione non si discosta concettualmente dalla consueta determinazione di stime statistiche in ambito multivariato, ma presenta caratteristiche particolari in termini di tempi di calcolo e di difficoltà di convergenza degli algoritmi standard di stima».

Una rete neurale artificiale del tipo feedforward è costituita normalmente di tre strati di unità elementari di calcolo (che chiamiamo nodi e hanno la stessa funzione che i neuroni hanno nelle reti neurali naturali), uno di input, uno intermedio e uno di output.

Un neurone artificiale funziona come segue: più attivazioni o segnali di input arrivano al neurone dall'esterno o da altri neuroni, diversamente pesate.

All'interno del neurone si sommano questi segnali di input diversamente pesati. Il risultato è trasformato con una funzione sigmoide, per lo più la logistica, che lo confina in un intervallo dato ed introduce un effetto soglia.

I neuroni di input raccolgono le informazioni da inviare ai neuroni dello strato intermedio, che a loro volta le trasformano tramite una funzione non lineare e inviano il risultato a tutti i nodi dello strato di output che ripetendo il procedimento forniscono il risultato finale.

Per ogni insieme di input una rete neurale artificiale deve produrre in uscita un valore.

Durante il processo di apprendimento si definiscono i pesi in base ai quali sono pesate le attivazioni. Il valore prodotto in uscita dalla rete sarà confrontato con un valore target (un insieme di dati di controllo). La rete stessa sarà quindi applicata a dati via via nuovi.

Le reti neurali artificiali sono strumenti importanti nell'ambito del connessionismo, termine che sintetizza l'idea di strutture costituite da unità capaci di elaborazioni, collegate da legami di intensità variabile che memorizzano le informazioni.

I sistemi connessionistici possiedono le seguenti proprietà:

- parallelismo: i carichi computazionali sono ripartiti tra nodi collegati da connessioni. «La presenza di legami in strutture parallele è comune a situazioni molto diverse tra loro: le unità di processo di una rete neurale artificiale; gli agenti umani in un sistema economico; gli esseri viventi in un sistema ecologico...», Terna (1994).

È il parallelismo che assicura la rapidità di reazione, di ragionamento, di riconoscimento e di classificazione proprie del nostro cervello, di per sé costituito di neuroni intrinsecamente lenti.

Operando simultaneamente si ottengono risultati che non dipendono strettamente, in termini di rapidità di calcolo, dalla limitatezza di ciascuno di essi;

- rappresentazione subsimbolica della conoscenza: Ogni legame tra nodi, ogni peso contribuisce con tutti gli altri a determinare direttamente e indirettamente i risultati.

È possibile intendere il concetto di rappresentazione subsimbolica della conoscenza in termini di rappresentazione distribuita della stessa;

- autorganizzazione: rappresenta una qualità assai diffusa nel mondo reale. Le strutture autorganizzanti sono comuni in campo naturale o sociale. Le strutture impersonali nel mondo economico reale, come il mercato, sono in realtà il frutto dell'azione di collettività di agenti autorganizzanti;

- robustezza all'errore

e

- ridondanza: sono caratteristiche che derivano principalmente dalla proprietà della rappresentazione subsimbolica della conoscenza, intesa come rappresentazione distribuita della conoscenza. In un sistema connessioneista errori nelle singole unità di processo non determinano il blocco del funzionamento, ma semplicemente un peggioramento dell'operatività.

Questo è dovuto al fatto che l'informazione è memorizzata in più pesi relativi a unità diverse. Ogni unità è responsabile solo in parte per il risultato finale.

2.1.10 La vita artificiale

L'idea di base della vita artificiale è quella di simulare grazie al computer l'esistenza di esseri virtuali che riproducendosi ed evolvendo possono aiutarci a chiarire alcuni aspetti riguardanti il perpetuarsi della vita ed il concetto di intelligenza.

All'interno della vita artificiale assumono particolare importanza gli algoritmi genetici ed i sistemi a classificazione.

Gli algoritmi genetici

Gli algoritmi genetici fanno parte del filone della programmazione genetica, inventata negli anni '80 dall'americano John Holland.

Il nome programmazione genetica deriva dal fatto che essa segue un principio simile a quello della selezione naturale: quando un essere vivente si riproduce crea figli non identici a se stesso, ma dotati di piccole mutazioni casuali. Le mutazioni che portano l'individuo ad essere meno adatto a vivere nel proprio ambiente scompaiono: l'individuo muore. Quelle che producono un vantaggio faranno sì che il loro possessore si riproduca più in fretta e con maggiore

successo. In questo modo le caratteristiche dell'individuo sono tramandate ai suoi figli.

Gli algoritmi genetici sono stati uno dei primi esempi di utilizzazione del paradigma evolutivo nella ricerca delle soluzioni di problemi.

Il metodo, sostiene Ferraris (2000), è basato sulla rappresentazione di ogni individuo come combinazione di due valori, zero e uno, in numero sufficiente a contenere tutta la conoscenza necessaria a contenere una qualche strategia. «Ad ogni individuo è associato un valore rappresentativo della sua *fitness*, calcolato in base ai risultati ottenuti applicando la strategia di cui è portatore.

Durante ogni iterazione, una certa quota della popolazione viene selezionata per la riproduzione, in modo da privilegiare gli individui con i valori di fitness più elevati; un'altra parte viene selezionata per l'estinzione.

La riproduzione avviene mediante copia ed incrocio delle stringhe genitori per simulare la compartecipazione al conferimento del patrimonio genetico dei discendenti. L'incrocio (crossover) si basa sull'individuazione, in modo casuale, di una posizione qualunque interna alla stringa, e sul successivo scambio delle parti a sinistra di quella fra i due individui coinvolti nell'operazione: in questo modo la conoscenza contenuta nei 'genitori' viene utilizzata per ottenere nuova conoscenza, rappresentata dai 'figli'».

Per simulare gli errori nella trasmissione del patrimonio genetico, continua Ferraris (2000), e per contribuire ad ampliare lo spazio della ricerca, viene applicata la mutazione: un'operazione che prevede l'inversione del valore di una o più posizioni prese a caso.

La presenza di un'elevata percentuale di stringhe uguali tra gli individui che costituiscono la popolazione, osservata dopo un congruo numero di riproduzioni, permette di «affermare che la strategia incorporata in quelle stringhe

risulta ‘buona’ per l’applicazione nell’ambiente dato».

I sistemi a classificatore

Per Goldberg (1989) un sistema a classificatore è «un sistema di apprendimento automatico che impara delle regole sintatticamente semplici (chiamate *classifier*) per guidare le sue azioni in un ambiente arbitrario».

All’interno dei sistemi a classificatore, sostiene Margarita (1992), si potrebbero ritrovare le caratteristiche e le limitazioni tipiche dei sistemi esperti basati su regole, se non fosse per alcune caratteristiche che li differenziano nettamente: l’attivazione parallela e non sequenziale delle regole, l’apprendimento automatico dell’importanza da attribuire ad ogni regola sulla base di un sistema competitivo e la creazione autonoma di regole tramite algoritmi genetici.

Gli elementi che compongono un sistema a classificatore sono i seguenti:

- un insieme di sequenze di lunghezza fissa che rappresentano le regole costruite sulla base di un alfabeto ternario;
- un sistema di sensori che riceve i messaggi dall’esterno e determina quali sono i classifier da questi attivati;
- un sistema di asta che determina quale dei classifier attivati agisce effettivamente ed esegue la propria azione;
- un sistema contabile che aggiorna il patrimonio dei singoli classifier sulla base dei premi percepiti in seguito alle decisioni prese;
- un algoritmo genetico per indurre nel sistema nuove sequenze, cioè nuove regole, ed eliminare le vecchie.

La dinamica dei sistemi a classificatore può essere suddivisa in diverse fasi:

- a fronte delle informazioni ricevute dall'ambiente, i sensori inviano uno o più messaggi ad una lista chiamata *message list*;
- tali messaggi, se soddisfano le condizioni codificate all'interno dei classifier, attivano questi ultimi;
- i classifier attivati diventano candidati all'invio di un proprio messaggio alla *message list*;
- i messaggi attivati possono attivare altri classifier oppure produrre un'azione rivolta all'esterno; i classifier che inviano un messaggio pagano un importo proporzionale al proprio patrimonio, che viene ripartito tra i classifier che lo hanno attivato, contribuendo così ad aumentarne il profitto.

La selezione dei classifier da attivare è basata sul patrimonio da essi posseduto, che diventa così equiparabile alla fitness degli algoritmi genetici.

2.1.11 L'intelligenza distribuita.

L'intelligenza artificiale distribuita si propone di studiare problemi complessi dividendoli in sottoproblemi più semplici e analizzando ciascuno di essi con un programma specifico che cooperi con gli altri programmi per arrivare alla soluzione finale.

Tra i primi a promuovere questo metodo fu Marvin Minsky, negli anni '70. Assieme a Seymour Papert, Minsky avanzò l'ipotesi che l'intelligenza non sia il prodotto di un meccanismo singolare, ma che abbia origine a partire dall'interazione di un grandissimo numero di agenti intelligenti.

Castelfranchi e Stock (2000) propongono un esempio: la percezione visiva della distanza, sostengono, è un compito banale per un bambino piccolo, ma

risulta quanto mai complicata per un robot.

Gestire questa funzione tramite l'intelligenza artificiale classica richiede un software gigantesco e terribilmente complesso.

La mente del bambino genera una mappa visiva utilizzando una serie di strutture neurali piccole e semplici, che comunicano e si coordinano, ma ognuna delle quali riesce in un compito semplice (percepire i bordi delle figure, la brillantezza di un oggetto visivo, ecc.).

L'intelligenza distribuita ha trovato applicazione in un gran numero di campi: dalle indagini di tipo economico al controllo del traffico aereo stradale, passando per i sistemi di monitoraggio ambientale e il riconoscimento della voce o di immagini.

Tra gli esperimenti portati avanti in questo campo è particolarmente interessante l'idea di Maja Mataric, riportata da Castelfranchi e Stock (2000), di combinare la creazione di una società di agenti, tipica dell'intelligenza artificiale distribuita, con l'idea della cibernetica, che prevede un'intelligenza reattiva, senza mente.

Mataric ha costruito una ventina di robot a partire da pezzi di automi e di calcolatori vecchi. Questi robot hanno il compito di trasportare da un capo all'altro del recinto dove risiedono una serie di dischi. I robot hanno alcuni riflessi e alcuni scopi istintivi: non urtare contro gli oggetti, non allontanarsi dal gruppo, non stare troppo vicini ad un altro robot, trasportare il maggior numero di dischi possibile.

Pur non essendo programmati per analizzare il mondo e per elaborare strategie, gli automi riescono ad agire in maniera intelligente, cooperativa, comunicando via radio e dividendosi i compiti.

2.1.12 È possibile l'intelligenza artificiale?

È possibile l'intelligenza artificiale?

Partendo dall'enunciato «Questa frase non è dimostrabile», ispirato al paradosso del mentitore, i logici hanno dimostrato che in ogni tipo di matematica, partendo da qualunque tipo di postulato possibile, ogni sistema formale è o inconsistente o incompleto.

Nel corso degli anni il teorema dell'incompletezza si è trasformato in un'arma contro l'idea di costruire delle macchine pensanti.

A partire da quanto detto, Lucas e Benacerraf propongono di immaginare, per assurdo, di costruire un algoritmo o una macchina complicata a piacere, che pretenda di simulare alla perfezione tutti i meccanismi del pensiero umano. Comunque sia realizzata la macchina si baserà su un sistema formale.

Noi umani possiamo analizzare tale sistema e scrivere, secondo le regole e i simboli del sistema stesso, la frase «Io non sono dimostrabile».

La macchina non potrà mai decidere manipolando i simboli se la frase è vera o falsa. Ma noi sì. Sappiamo grazie all'intuizione logica che la frase è vera, perché se fosse falsa il sistema sarebbe incoerente, assurdo. Poiché una macchina non sarebbe in grado di mostrare la verità della frase, mentre la mente umana sì, non possiamo considerare la macchina un modello adeguato e completo della mente.

L'obiezione principale a questo tipo di tesi è semplice: per il teorema di Gödel, però, questa frase esiste solo se il sistema è consistente; ma chi ci assicura che un algoritmo così complesso da ricreare la coscienza sia consistente? Chi ci assicura che la mente umana stessa sia consistente e che in caso affermativo non soffrirebbe delle stesse limitazioni dei computer?

2.1.13 L'enigma della stanza cinese.

Un argomento anti-intelligenza artificiale più semplice è stato proposto dal filosofo americano John Searle. Searle non è contrario in assoluto all'idea di intelligenza artificiale, ma è convinto che solo macchine particolarissime che possiedono gli stessi poteri causali del cervello possano pensare.

Ciò che Searle rifiuta è la versione forte dell'intelligenza artificiale, quella per cui pensiero e coscienza possono essere racchiusi in un algoritmo e ricreati su una macchina di Turing che manipoli simboli.

Secondo Searle è assurdo pensare di aver ricreato la mente solo perché si è scritto un programma per calcolatore che reagisce alle nostre domande come farebbe una persona.

Searle propone un esperimento mentale per mostrare l'incoerenza logica della tesi dell'intelligenza artificiale forte e del test di Turing: l'enigma della stanza cinese.

«Si consideri una lingua che un individuo non conosce, dice Searle (1990), io ,per esempio, non conosco il cinese». Supponiamo che esista un algoritmo tanto perfetto da poter superare il test di Turing in lingua cinese: un cinese che comunicasse con il programma avrebbe l'impressione di avere a che fare con un'altra persona di madrelingua cinese. Immaginiamo di stampare tutte le istruzioni di tale programma in un unico manuale e di rinchiuderci con esso in una stanza chiusa, comunicante con l'esterno solo tramite una buca delle lettere.

Se qualcuno ci inviasse dall'esterno una frase in cinese, per esempio «Qual è il tuo colore preferito?», ai nostri occhi questa apparirebbe come una sequenza incomprensibile di ideogrammi.

Seguendo le regole del manuale, tuttavia, saremmo in grado di mettere insieme una sequenza di simboli per rispondere alla domanda. Nonostante questo

non capiremmo nulla di cinese, poiché le regole del manuale non traducono la lingua, ma richiedono semplicemente di rispondere in maniera meccanica ad una sequenza di simboli con un'altra sequenza di simboli.

La persona che ci ha rivolto la domanda, tuttavia, riceverà una risposta in perfetto cinese e sarà convinto di aver comunicato veramente con una persona di lingua cinese.

Searle conclude dicendo che in un sistema come quello descritto sarebbe impossibile capire il cinese, perché non si potrebbe imparare il significato dei simboli.

Un calcolatore manipola simboli, ma questo non è necessario a garantire l'intelligenza. Il calcolatore è dotato di una sintassi, ma non conosce una semantica, cioè non associa e non può associare nessun significato ai simboli che manipola.

Il cervello, inoltre, è dotato di poteri causali, oltre che manipolare pensieri e simboli è in grado di causare nuovi pensieri e nuove azioni. Un computer programmato no.

Questa argomentazione, per alcuni ricercatori, ha posto fine ad un'idea di intelligenza artificiale intesa in senso forte, tanto che Roberto Satolli (1999) ambienta nella stanza cinese un giallo senza assassino, quello della signora Godai (acronimo di Good Old Fashioned Artificial Intelligence, l'intelligenza artificiale fatta alla buona vecchia maniera), morta suicida.

A Searle fa eco Parisi (1998), che sostiene che l'uomo arriverà probabilmente a costruire una macchina pensante, ma che per farlo bisognerà riprodurre le proprietà fisiche degli esseri umani.

Molti altri ricercatori sono invece rimasti fedeli all'intelligenza artificiale forte e hanno criticato duramente il paradosso della stanza cinese.

Douglas Hofstadter sostiene che è nel software, nel manuale, che si nasconde

la mente, la comprensione del cinese, non nella persona (l'hardware) che dovrebbe eseguire le istruzioni.

Ma la contromossa strategica più nota ai sostenitori dell'intelligenza artificiale è paradossale: «l'argomento di Searle è sbagliato semplicemente perché, per assurdo che sembri a prima vista, anche se il signore nella stanza non capisce il cinese, la stanza nel suo complesso sì. È un ente che pensa e comunica in cinese!», Castelfranchi e Stock (2000).

Capitolo 3

La complessità.

3.1 Complessità: la scienza del ventunesimo secolo

3.1.1 La teoria del Caos

Il portale sapere.it definisce la teoria del caos

[una] recente branca della scienza che studia, attraverso modelli computerizzati, la ricorsività nei sistemi complessi [...] caratterizzati da un così alto grado di irregolarità da non poter essere determinata con il calcolo tradizionale.

La gran parte dei sistemi dinamici è soggetta al caos. L'evoluzione di tali sistemi dipende in modo sensibile dalle condizioni iniziali, quali che esse siano. Questo significa che un piccolo mutamento nello stato del sistema al tempo zero produce un mutamento ulteriore che cresce in modo esponenziale con il tempo, una piccola causa ha un grande effetto. L'evoluzione di questi sistemi conduce ad un comportamento essenzialmente imprevedibile e casuale.

Il primo ad interessarsi a questi fenomeni fu Henri Poincaré, ricercatore francese della seconda metà dell'Ottocento, Poincaré notò che:

Una causa molto piccola che sfugge alla nostra attenzione determina un effetto considerevole che non possiamo fare a meno di notare, e allora diciamo che questo effetto è dovuto al caso. Se conoscessimo esattamente le leggi di natura e lo stato dell'universo nel momento iniziale, potremmo predire esattamente lo stato dello stesso universo in un momento successivo. Ma se fosse anche il caso che le leggi naturali non avessero più segreti per noi, potremmo tuttavia conoscere solo approssimativamente lo stato iniziale. [...] può accadere che piccole differenze nelle condizioni iniziali [ne] producano di molto grandi nei fenomeni finali. Un piccolo errore nelle prime produrrà un errore enorme nei secondi. La predizione diviene impossibile, e abbiamo il fenomeno fortuito, Ruelle (1991).

Questa presa di posizione si scontra con il determinismo rigoroso che caratterizzava la società in cui Poincaré viveva, determinismo frutto della visione meccanicista della scienza newtoniana.

«La gigantesca macchina cosmica era considerata completamente causale e determinata. Tutto ciò che avveniva aveva una causa definita e dava luogo a un effetto definito e, in linea di principio, si sarebbe potuto prevedere con assoluta certezza il futuro di una qualsiasi parte del sistema se si fosse conosciuto in un qualsiasi istante il suo stato in tutti i suoi particolari», Capra (1975).

Questa convinzione trovò la sua espressione più chiara nelle famose parole del matematico francese Pierre-Simon de Laplace:

Un'intelligenza che, a un dato istante conoscesse tutte le forze

da cui è animata la natura e la situazione rispettiva degli esseri che la compongono, se per di più fosse abbastanza profonda da sottomettere questi dati all'analisi, abbraccerebbe nella stessa formula i movimenti dei più grandi corpi dell'universo e dell'atomo più leggero: nulla sarebbe incerto per essa e l'avvenire, come il passato, sarebbe presente ai suoi occhi, Capek (1967).

La teoria del caos ha un secondo padre, Edward Lorenz, che nel 1961 si occupava di formalizzare un sistema di equazioni per la previsione dei fenomeni meteorologici tramite computer.

Un giorno, volendo mettere alla prova le sue equazioni, Lorenz introdusse nel calcolatore una rilevazione approssimata alla terza cifra decimale, invece che alla sesta.

Il calcolatore effettuò una previsione completamente diversa da quella che avrebbe dovuto formulare. Lorenz capì così che un sistema dinamico estremamente complesso, come quello che lega le variabili che entrano in gioco nel determinare le condizioni meteorologiche, è estremamente sensibile alle condizioni iniziali. Una differenza anche minima di queste e lo stato del sistema può evolversi in maniera completamente diversa.

Questo fenomeno è noto con il nome di Butterfly effect (effetto farfalla), poiché, partendo da questo ragionamento, è possibile affermare che il battito delle ali di una farfalla in Cina può provocare un tornado all'altro capo del mondo.

Il comportamento dei sistemi caotici, sostiene Ruelle (1991), non è intrinsecamente indeterministico: è matematicamente dimostrabile che le condizioni iniziali sono sufficienti a fissare l'intero comportamento futuro del sistema in maniera esatta e univoca. Il problema insorge quando si cerca di specificare le condizioni iniziali.



Figura 3.1: Lo strano attrattore di Lorenz

Dal momento che in pratica non si può mai conoscere con assoluta precisione lo stato iniziale di un sistema, l'errore che si compie nell'osservazione produrrà effetti nei confronti delle previsioni.

Una seconda idea che è normalmente associata alla teoria del caos, Barkley Rosser Jr. (1999), è che molti sistemi caotici tendono a seguire traiettorie che formano complicate e irregolari figure geometriche note come strani attrattori.

Un esempio interessante di strano attrattore è l'attrattore di Lorenz.

Lorenz era un meteorologo, Ruelle (1991), e si occupava del problema della convezione atmosferica, ovvero di quell'insieme di movimenti che derivano dal riscaldamento del suolo da parte del sole: gli strati inferiori dell'atmosfera divengono più caldi e più leggeri degli strati più elevati, determinando conseguentemente un moto ascendente dell'aria calda, più leggera, mentre l'aria fredda, più densa, discende.

L'attrattore di Lorenz è un insieme complesso in cui ha luogo un'evoluzione temporale in tre dimensioni.

Lorenz effettuando un'approssimazione si limita a studiare l'aria, che essendo un fluido andrebbe rappresentata come un punto in uno spazio infinito e tridimensionale mediante tre equazioni differenziali non lineari.

Immaginando lo stato dell'atmosfera in convezione come rappresentato da un punto P, lo spostamento temporale ha luogo secondo una linea curva. Il punto P gira attorno ad una delle orbite dell'attrattore, poi più volte attorno all'altra, per tornare a girare attorno alla prima. L'orbita è limitata, ma non è periodica né tanto meno convergente. Se la posizione iniziale di P venisse modificata anche lievemente i particolari dell'evoluzione temporale ne risulterebbero completamente modificati. L'andamento tendenziale (l'aspetto generale della figura), sarebbe il medesimo, ma la sua generazione e il numero e la sequenza di giri compiuti da P attorno all'attrattore sarebbero completamente diversi.

Questo fenomeno è dovuto al fatto che l'evoluzione temporale del sistema dipende in modo sensibile dalle condizioni iniziali. Ne consegue l'imprevedibilità, a lungo termine, delle condizioni atmosferiche.

3.1.2 La formica di Langton

La visione laplaciana di un universo ordinato e prevedibile appare ormai ingenua. «Leggi semplici possono generare comportamenti estremamente complessi, e i sistemi deterministici possono comportarsi in modo casuale», Ian Stewart (1994). La formica di Langton costituisce un ulteriore esempio utile per illustrare questo concetto.

La formica di Langton è un automa cellulare inventato da Chris Langton, ricercatore al Santa Fe institute.

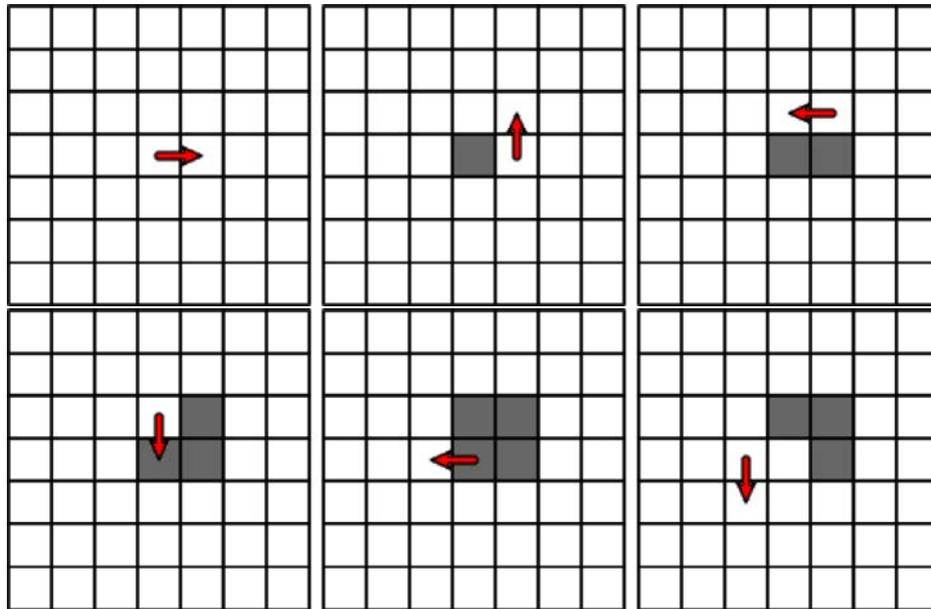


Figura 3.2: I primi passi della formica di Langton.

La formica parte dal quadrato centrale di un reticolo e punta in una direzione particolare, per esempio verso est, spiega Ian Stewart (1994). si sposta di un quadrato in quella direzione e prende in considerazione il colore del quadrato su cui è venuta a trovarsi, bianco o nero. Se si trova su un quadrato nero, lo colora di bianco e gira di novanta gradi a sinistra. Se si trova su un quadrato bianco, lo colora di nero e gira di novanta gradi a destra. Va avanti all'infinito seguendo queste semplici regole.

Ipotizziamo di partire da un reticolo perfettamente bianco. Per i primi cinquecento passi circa, la formica continua a tornare al quadrato centrale, lasciando dietro di sé una serie di configurazioni abbastanza simmetriche. Per i successivi 10000 passi circa, l'immagine diviene molto caotica. All'improvviso la formica costruisce un'autostrada¹.

Segue più volte una successione di esattamente centoquattro passi che la

¹La definizione è di James Propp, del Massachusetts Institute of Technology

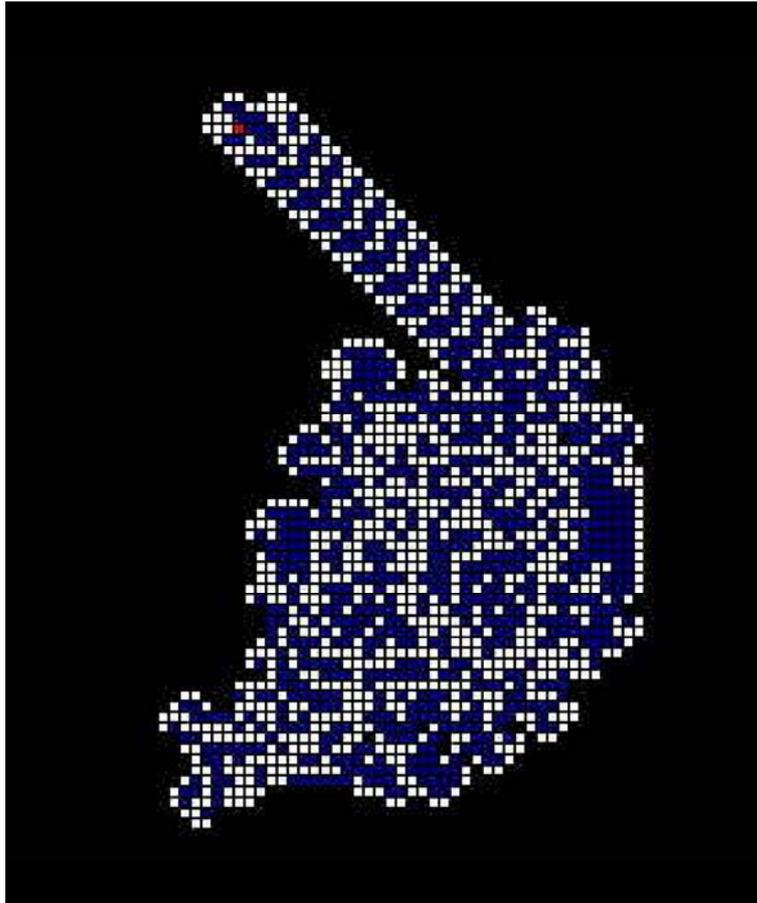


Figura 3.3: La formica di Langton al passo 11111. La formica sta costruendo la sua 'autostrada'

fanno muovere di due caselle verso nord-ovest e continua così indefinitamente formando una striscia diagonale.

Conosciamo le regole che sono alla base del movimento della formica di Langton. Ci troviamo nella situazione descritta da Laplace. Sappiamo che la tendenza della formica sembra essere quella di costruire sempre un'autostrada. Tuttavia non siamo in grado di determinare se partendo da un ambiente arbitrario, non più tutto bianco, essa costruirà sempre un'autostrada.

Esistono moltissime generalizzazioni, continua Ian Stewart (1994), che mostrano comportamenti sorprendenti e configurazioni comuni. ci si può divertire a mettere una o più formiche in un ambiente prescelto e a osservare il loro comportamento. Si possono cambiare le regole e predisporre ambienti diversi.

Sono state anche studiate delle formiche generalizzate definite da una stringa di regole.

Supponiamo che, anziché essere soltanto bianchi o neri, i quadrati possano essere di n colori, contrassegnati con $=, 1, 2 \dots n-1$. La stringa di regole è una successione di n simboli 0 o 1. Quando una formica lascia una casella di colore k , la cambia attribuendole il colore $k+1$ (con $n=(n-1)+1$ che torna a 0).

Svolta a destra se il k -esimo simbolo è 1 e a sinistra se è 0. Si sposta di un quadrato e ripete.

«Le regole originali di Langton si riassumono nella stringa di regole 10. Alcune stringhe di regole portano la formica ad un comportamento banale: per esempio una formica con stringa di regole 1 (o anche 111 ... 1) si sposta indefinitamente intorno ad un quadrato 2×2 . Ma qualsiasi stringa di regole che contenga sia 0 sia 1 deve portare a traiettorie illimitate», Ian Stewart (1994).

Supponiamo per semplicità di partire da una griglia pulita, in cui tutte le caselle abbiano colore 0. La formica 100 crea configurazioni che inizialmente somigliano abbastanza a quelle della formica di Langton: dapprima simmetriche e poi caotiche. Dopo 150 milioni di passi continua a comportarsi in modo caotico. È impossibile sapere se costruirà mai un'autostrada.

La formica 110 costruisce un'autostrada e impiega solo centocinquanta passi per farlo.

La formica 1000 è altamente caotica. Quella 1101 inizia in modo caotico, ma avvia la costruzione di un'autostrada dopo 250000 passi.

La formica 1100 costruisce configurazioni via via più complesse che hanno simmetria bilaterale e quindi non può costruire nessuna autostrada nel senso usuale del termine.

È difficile credere che si possa trovare una semplice classificazione dei comportamenti di tutte queste formiche generalizzate o a prevedere dalla loro stringa di regole quale sarà il loro comportamento a lungo termine.

«È bene notare che regole di transizione appena più complesse portano a esempi come il gioco «Vita» di John Horton Conway», Ian Stewart (1994). Conway ha dimostrato che in vita esistono configurazioni che formano macchine di Turing universali e quindi dal comportamento a lungo termine indecidibile. Il problema dell'arresto, tradotto nei termini del gioco «Vita» assume la forma della domanda 'Questa configurazione cresce illimitatamente?', frase formalmente indecidibile.

Questa è un'ulteriore risposta all'assunto laplaciano per il quale possedere le leggi che governano l'andamento dell'universo dovrebbe renderci in grado di poter prevedere in qualsiasi momento lo stato dell'universo stesso.

3.1.3 La definizione di complessità.

Barkley Rosser Jr. (1999) sostiene che non esiste una definizione di ‘complessità’ che riesca a mettere tutti d’accordo. Secondo Richard H. Day (1994) «un sistema dinamico è complesso se endogenamente non tende asintoticamente ad un punto fisso, a un ciclo limite o ad una esplosione».

Questi sistemi possono mostrare un comportamento discontinuo e possono essere descritti tramite insiemi di equazioni non lineari differenziali, Richard H. Day (1994). È bene notare però che non tutti i sistemi di equazioni di questo tipo genereranno fenomeni complessi.

La definizione di complessità fornita da Day non abbraccia la totalità dei fenomeni che gli economisti chiamano ‘complessi’, in realtà, in dottrina, si registrano differenti opinioni a proposito di che cosa voglia dire complesso.

Stodder (1997) e Albin e Foley (1998) sostengono che una situazione ha natura complessa quando sussiste un’estrema difficoltà nel calcolare le soluzioni relative ad un problema di ottimizzazione.

Pryor (1995) e Stodder (1995) si concentrano invece su un’ottica strutturale. Per loro complessità è sinonimo di esistenza di un gran numero di complicate interrelazioni e di strutture istituzionali all’interno dell’economia.

Pietro Terna (2001) sottolinea come il concetto di complesso debba essere separato da quello di complicato e propone il seguente esempio. «Un motore a scoppio è certamente molto complicato, ma smontandolo riusciamo a comprendere come ciascuna sua parte interviene nel sistema, di cui afferriamo molto bene il funzionamento; un formicaio è un sistema complesso, il cui funzionamento è difficile da comprendere; soprattutto, l’esame isolato delle diverse componenti (i diversi tipi di formiche) ci dice pochissimo sul ruolo delle diverse parti e sulla meccanica del sistema», Terna (2001). Per comprendere il formicaio, quindi, è necessario studiare allo stesso momento

le sue componenti (le formiche) ed il sistema aggregato che ne deriva.

Arthur, Durlauf e Lane (1997) propongono sei caratteristiche che dovrebbero servire a definire cosa è complessità:

1. Interazione dispersa tra agenti eterogenei che agiscono localmente tra di loro in uno spazio.
2. Nessuna possibilità per un singolo agente di poter monopolizzare tutte le risorse dell'economia, anche in caso di interazioni globali deboli.
3. Organizzazioni gerarchiche che si intersecano.
4. Adattamento continuo degli agenti.
5. Innovazione continua per quanto riguarda mercati, tecnologie, comportamenti e istituzioni.
6. Dinamiche prive di un unico equilibrio che portano il sistema a modificarsi nel tempo restando lontano da un ottimo globale.

Anche Parisi (2001) propone una distinzione tra sistemi semplici e sistemi complessi. I sistemi semplici hanno le seguenti caratteristiche:

- sono lineari;
- permettono di conoscere gli stati successivi del sistema sulla base degli stati precedenti;
- si trasformano nel tempo in modo prevedibile;
- sono perturbati da un evento esterno in maniera commisurata all'entità della perturbazione (una perturbazione piccola produrrà effetti modesti, una perturbazione grande ne produrrà di ingenti);

- un sistema semplice può essere isolato dal contesto;
- un sistema semplice è composto di parti che hanno un ruolo facilmente individuabile nell'influenzare il comportamento complessivo del sistema stesso;
- un sistema semplice può essere riprodotto in copie identiche.

Le caratteristiche di un sistema complesso sono invece le seguenti:

- gli stati futuri di un sistema complesso non sono in genere prevedibili a partire dagli stati precedenti;
- le trasformazioni nel tempo di un sistema complesso sono anch'esse difficilmente prevedibili;
- le perturbazioni esterne che agiscono su un sistema complesso producono sul sistema stesso effetti che tendono a essere non commisurati all'entità della perturbazione;
- i sistemi complessi sono molto sensibili alle loro condizioni iniziali: due sistemi complessi che partono da condizioni iniziali appena differenti possono svilupparsi nel tempo in modo molto diverso;
- i sistemi complessi sono sensibili al contesto in cui operano;
- i sistemi complessi tendono ad essere coinvolti in rapporti di causa reciproca;
- i sistemi complessi tendono ad essere coinvolti in gerarchie di sistemi dove un sistema a un certo livello costituisce un elemento di un sistema più grande al livello immediatamente successivo della gerarchia;

- non è facile identificare il ruolo che ogni elemento di un sistema complesso ha nel determinare il comportamento del sistema stesso.

Barkley Rosser Jr. (1999) identifica quattro correnti di studio sulla non linearità che hanno portato allo studio attuale dei sistemi complessi:

1. La cibernetica. Forrester (1961) ipotizza che all'interno di sistemi multipli di equazioni non lineari possano emergere risultati controintuitivi. Questa idea è un principio fondamentale degli studi sui modelli dinamici nei quali è presente il concetto di auto-organizzazione e di emergenza di strutture organiche dall'interazione di sistemi semplici.
2. La teoria delle catastrofi. René Thom (1975) sviluppò la teoria delle catastrofi a partire dalla teoria dei sistemi dinamici. Una catastrofe è un tipo particolare di discontinuità in un sistema. Le discontinuità sono dovute all'esistenza di differenti punti di equilibrio all'interno del sistema. Al variare dei parametri si assiste ad un salto da un punto di equilibrio all'altro.
3. La teoria del caos. Per la teoria del caos le dinamiche caotiche sono frutto di un processo deterministico sebbene il loro comportamento appaia in prima analisi casuale.
4. La complessità. La complessità accoglie ed evolve tutte le premesse teoriche delle precedenti correnti di pensiero.

I sistemi complessi adattivi

Holland (1975) definisce i sistemi complessi come

«gruppi di agenti legati in un processo di co-adattamento, in cui le

mosse di adattamento di ciascuno hanno conseguenze per l'intero gruppo di individui».

I sistemi complessi, continua Holland (1975), sono strettamente correlati con i sistemi non lineari. Un sistema non lineare è

«un sistema il cui comportamento non è uguale alla somma delle parti che lo caratterizzano».

Per studiare i sistemi lineari è sufficiente procedere alla scomposizione e allo studio delle loro componenti. Lo stesso meccanismo non può essere messo in atto per quanto riguarda lo studio dei sistemi non lineari.

Un sistema complesso è definito adattivo se è costituito da agenti che modificano le loro azioni in funzione di eventi che si scatenano durante il processo di interazione.

Holland (1995) sostiene che l'analisi dei sistemi adattivi complessi deve essere condotta a due livelli:

- Lo studio dei modelli interni

e

- Lo studio dei building block.

I modelli interni, nei sistemi adattivi complessi, si identificano con i meccanismi che generano il fenomeno dell'anticipazione.

Questo elemento ci permette di interpretare al meglio i meccanismi correlati con l'adattamento e con la capacità di apprendimento che si osservano ai diversi livelli di complessità. Questi meccanismi dipendono dall'esperienza e dalla sensibilità di ciascun agente.

Abbiamo due tipi di modelli interni: quelli impliciti e quelli espliciti.

I primi condizionano le azioni degli agenti sulla base di un'implicita previsione di alcuni eventi futuri. I secondi comportano un'esplorazione interna delle alternative (processo di lookahead).

La struttura dell'ambiente contribuisce a determinare il comportamento del singolo agente. L'agente possiede un modello interno se le azioni intraprese sono frutto di anticipazione delle conseguenze future. Quando esiste un meccanismo che consente di mettere efficacemente in relazione un'azione corrente e le sue conseguenze future, l'evoluzione può favorire i modelli interni efficienti ed eliminare quelli inefficienti.

Il secondo livello di analisi è costituito dai building block, che Holland (1995) descrive come elementi riutilizzabili e ripetitivi di un sistema. I building block permettono di costruire gerarchie senza spendere troppe risorse. Un esempio di building block sono i neuroni all'interno di una rete neurale o gli oggetti in una rappresentazione visiva.

I modelli interni danno forma ai building block.

Nelle situazioni reali un modello interno deve essere fondato su un numero limitato di esempi relativi ad un ambiente in continuo mutamento. Il modello è utile solo se è possibile trovare elementi comuni all'interno delle differenti situazioni descritte.

È necessario, quindi, individuare una metodologia capace di individuare questi elementi comuni.

Holland (1995) sostiene che l'uso dei building block è particolarmente efficace quando questi sono utilizzati in progetti che comportano processi di aggregazione, cioè processi che riguardano l'emergenza di comportamenti complessi su larga scala dalle interazioni aggregate di un numero inferiore di agenti complessi.

Merry (1999) fornisce una descrizione degli elementi che caratterizzano in

sistemi adattivi complessi:

1. Ogni sistema adattivo complesso è composto da una rete di componenti che costantemente e reciprocamente si influenzano tra di loro.
2. Il sistema è decentrato (nessuna componente è in grado di controllare le altre. I comportamenti degli agenti, apparentemente coerenti, sono frutto della competizione e della collaborazione tra le componenti
3. Ogni livello rappresenta un building block per il livello successivo.
4. Il meccanismo fondamentale è l'adattamento, che si realizza in seguito alla costante riorganizzazione interna delle componenti.
5. I sistemi adattivi complessi sono in grado di prevedere gli accadimenti futuri. Le previsioni sono basate sui cambiamenti che avvengono nei modelli interni (la rappresentazione che ogni agente ha della realtà in cui esso opera). Questi modelli sono costantemente modificati grazie all'esperienza.
6. Le opportunità sono create dal sistema stesso che sviluppa costantemente nuovi fenomeni e nuove situazioni.
7. Ogni processo è in costante cambiamento. Non si raggiunge mai un equilibrio ottimale perché esiste un continuo flusso di relazioni con gli altri sistemi che provocano perturbazioni.

Le formiche

Le formiche costituiscono un argomento di studio molto interessante per gli studiosi della complessità. Questi animali dimostrano come l'interagire di un gran numero di agenti in maniera apparentemente casuale possa far emergere

comportamenti ordinati (basti pensare che il formicaio, l'organismo collettivo di cui le formiche fanno parte, è termoregolato e la sua temperatura tra estate e inverno cambia di appena un grado).

Un esperimento significativo per arrivare a comprendere meglio il comportamento delle formiche è stato compiuto da alcuni entomologi, Deneubourg et al. (1987) e Pasteels et al. (1987).

Due fonti di cibo sono state poste ad eguale distanza da un formicaio. Le due fonti erano entrambe strutturate in modo da contenere sempre il medesimo livello di cibo. Dopo un certo periodo le formiche individuavano e cominciavano a sfruttare le due fonti.

Ciò che più colpì i ricercatori fu che le formiche, nel lungo periodo, invece di dividersi equamente tra le due fonti, ne sfruttavano una in maniera preferenziale. In effetti l'80 % delle formiche impegnate nella raccolta di cibo sfruttava una particolare fonte, il restante 20 % l'altra.

Per essere sicuri che la differenza di disposizione delle formiche non fosse dovuta ad una qualche impercettibile differenza tra le due fonti, Pasteels et al. ripeterono l'esperimento eliminando una delle fonti, ma rendendo la restante accessibile solo passando attraverso a due ponti simmetrici.

I risultati del primo esperimento furono confermati.

Pasteels et al. omisero però un elemento cruciale per capire il comportamento delle formiche², sottolinea Kirman (1993).

Durante l'esperimento avvenivano periodici 'capovolgimenti di fronte' che portavano le formiche ad invertire le loro preferenze. Il ponte che era prima attraversato dall'80 % delle formiche passava ad ospitare il 20 % delle stesse, quello meno transitato accresceva il numero dei suoi 'utenti' fino a ospitare l'80 % delle formiche impegnate in operazioni di raccolta del cibo.

²Kirman riferisce di aver ricevuto personalmente l'informazione da J. Pasteels.

Le formiche, quando vanno a caccia di cibo, si dispongono sul territorio seguendo traiettorie per nulla regolari che hanno il solo scopo di permettere alla formica di ‘pettinare’ quanto più territorio possibile.

Quando una formica individua una fonte di cibo è in grado di portarci altre formiche.

Per farlo l’insetto torna al formicaio e stimola fisicamente, tramite contatto o con una secrezione, un’altra formica affinché essa lo segua.

In alternativa esso può lasciare una traccia chimica (feromoni) che le altre formiche possono seguire fino al luogo dove è stato individuato il cibo.

Poiché anche la fonte più grande di cibo è destinata ad esaurirsi prima o poi, le formiche non hanno il solo problema di massimizzare il benessere del formicaio (e quindi il flusso di cibo che entra all’interno di quest’ultimo), ma anche quello di far sì che questo flusso non scenda mai sotto ad una soglia minima.

Così, per garantire la sopravvivenza della colonia, è utile che qualche formica vada sempre a caccia di nuove fonti di cibo.

Quando una formica diretta alla prima fonte di cibo (fonte A) ne incontra un’altra diretta alla seconda fonte di cibo (fonte B) una delle due si lascia ‘convincere’ e procede con la seconda formica verso la destinazione originaria di quest’ultima.

Poniamo il caso che entrambe le formiche si siano recate alla fonte B. L’aumento delle formiche destinate alla fonte B (formiche B) aumenta la probabilità che un’ulteriore formica incontri una formica B e si diriga a sua volta verso la fonte B.

Questo processo farà sì che in breve tempo gli insetti B prevarranno in numero seguendo un meccanismo di retroazione positiva (positive feedback). La ragione del prevalere di B su A (o di A su B) è dovuta a microfenomeni

casuali presenti nel sistema al tempo zero.

Questi fenomeni, grazie al meccanismo della retroazione positiva, fanno sì che nel sistema si creino squilibri che hanno la tendenza a permanere. Quando un sistema si ‘blocca’ assistiamo ad un fenomeno di lock-in.

Kirman (1993) propone una semplice spiegazione per lo strano comportamento degli insetti: una formica può assumere in ogni momento tre diversi comportamenti: può rifornirsi dalla fonte di cibo alla quale ha attinto fino al momento, può cambiare fonte di cibo per volontà propria o può essere convinta a fare lo stesso da un'altra formica.

Partiamo dall'ipotesi che la probabilità che la formica cambi idea in maniera autonoma sia molto bassa.

Per Kirman, il sistema nel suo complesso non possiede un punto di equilibrio stabile, ma passa continuamente da uno stato all'altro. La velocità con cui tale cambiamento avviene dipende direttamente dal grado di influenzabilità che i singoli componenti del sistema (le formiche) possiedono.

Controintuitivamente, più le formiche sono propense a cambiare idea più il sistema, nel suo complesso, sarà composto di situazioni in cui le formiche si dividono tra le due fonti di cibo secondo proporzioni che si avvicinano al 50%.

Più le formiche saranno convinte delle loro scelte, invece, più grandi saranno le quantità di tempo in cui il sistema resta bloccato in corrispondenza di forti situazioni di squilibrio.

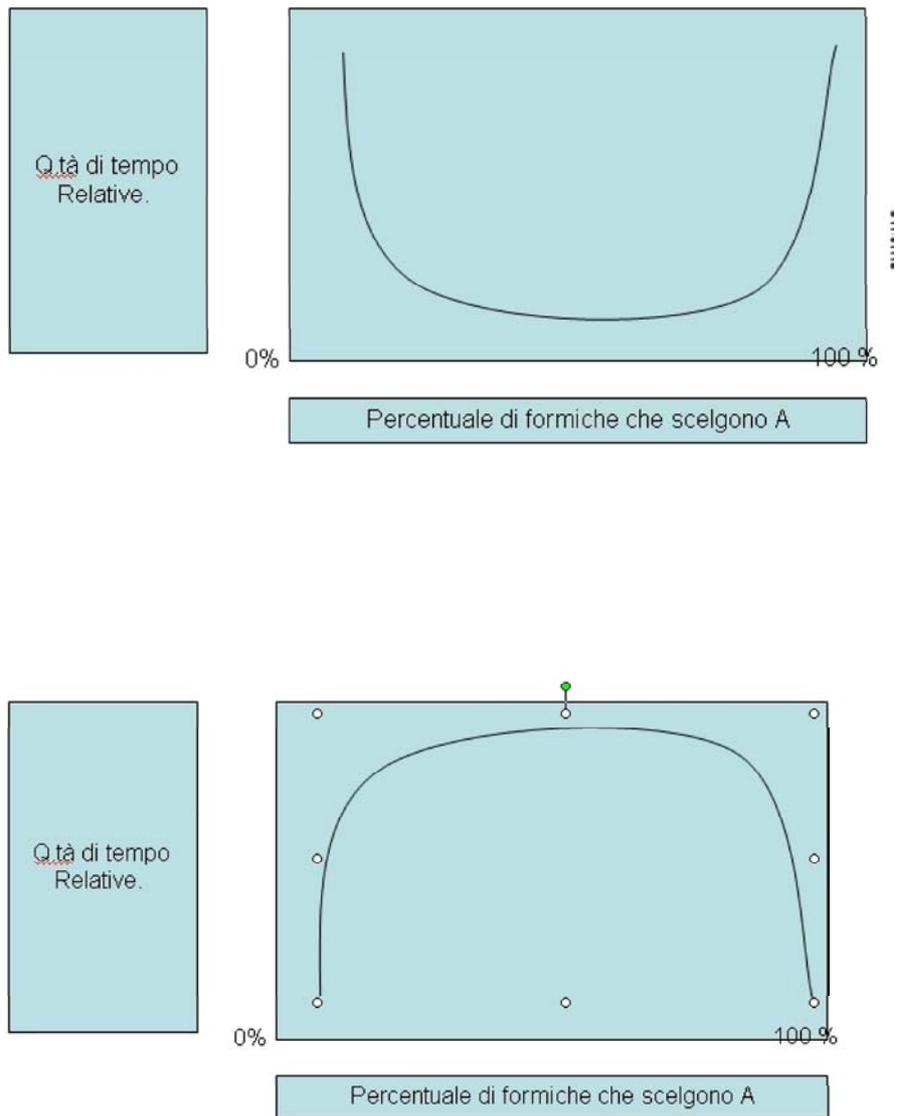


Figura 3.4: Bassa e alta propensione a cambiare comportamento.

A partire dal lavoro di Kirman, Paul Omerod (2003) ha sviluppato una serie di esempi importanti nell'ambito dell'economia e delle scienze sociali. In questi esempi i protagonisti, come le formiche di Kirman, hanno la possibilità di seguire sempre tre diversi comportamenti: possono agire coerentemente con le loro decisioni passate, possono farsi influenzare dalle decisioni altrui e possono infine cambiare idea autonomamente.

Posti di fronte ai più diversi problemi, le 'formiche' di Omerod forniscono spiegazioni interessanti a quesiti che vanno dall'individuare la capacità deterrente sui potenziali delinquenti dell'inasprimento delle pene carcerarie al perché alcuni film e alcuni oggetti diventano di gran moda e altri, magari migliori, si rivelano un totale fallimento.

In tutti gli esempi proposti Omerod sottolinea che la comunicazione tra i diversi agenti, la 'vita sociale', ha importanza centrale nel processo di scelta, che la decisione riguardi cosa andare a vedere al cinema o se commettere o meno un delitto.

Un altro aspetto che Omerod vuole mettere in evidenza è che le azioni compiute dai protagonisti degli esempi tendono a formare dinamiche non lineari. Il livello di variabilità del tasso di criminalità in una popolazione in funzione della severità del sistema penale ne è un esempio.

Omerod (2003) presenta il seguente grafico:

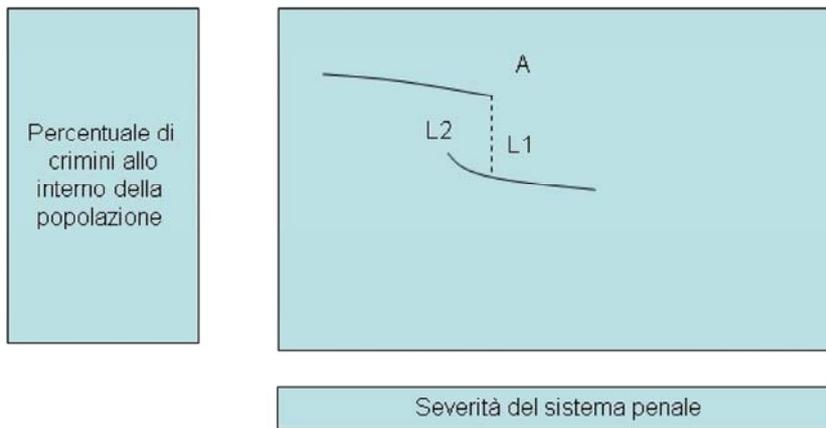


Figura 3.5: Percentuale di crimi ni all'interno della popolazione in funzione severità del sistema penale

«L'irrigidimento del sistema penale opera come deterrente della criminalità e la percentuale di crimini nell'intera popolazione diminuisce», Omerod (2003). Inizialmente, tuttavia, come possiamo osservare sulla curva superiore del grafico, l'effetto è molto modesto. Spostandoci lungo la curva da sinistra verso destra notiamo che per incrementi uguali della severità degli incentivi negativi, gli effetti sulla criminalità risultano progressivamente maggiori.

In corrispondenza del punto critico A, il sistema si sposta bruscamente al livello L1 situato sulla curva inferiore.

«Un ricercatore che fosse in possesso unicamente dei dati relativi ad alcuni punti all'immediata sinistra del punto A sulla curva superiore e di tutti i dati a sinistra di tale punto sulla curva inferiore potrebbe solo concludere che la maggior brevità delle condanne sembra ridurre la criminalità», Omerod (2003).

Anche per questa ragione la letteratura convenzionale sulla criminalità non giunge a conclusioni unanimi in merito agli effetti delle politiche anticrimine.

Un discorso analogo va fatto, aggiunge Omerod (2003), per quanto riguarda il rapporto tra la percentuale di crimini commessi all'interno di una popolazione e il livello di deprivazione sociale ed economica della popolazione stessa.

Il grafico seguente è relativo a quest'ultima situazione:

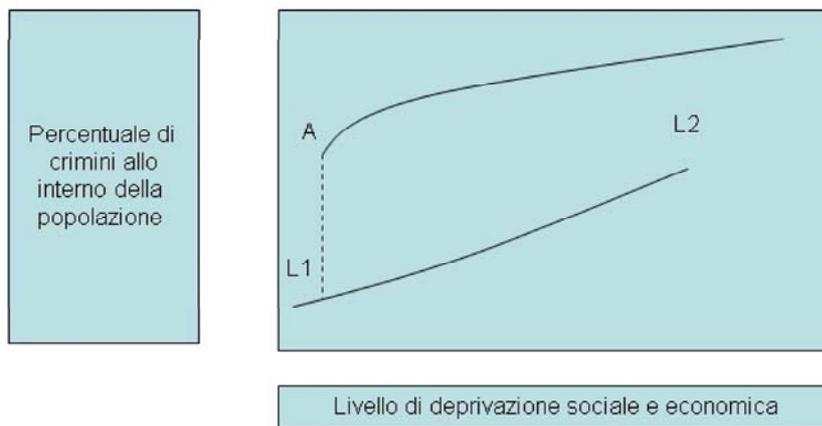


Figura 3.6: Percentuale di crimi ni all'interno della popolazione in funzione della deprivazione sociale ed economica.

Capitolo 4

La simulazione.

También el jugador es prisionero
(La sentencia es de Omar) de otro tablero
de negras noches y de blancos días.

Dios mueve al jugador y éste, la pieza.
¿Qué dios detrás de Dios la trama empieza
De polvo y tiempo y sueño y agonía?

Anche il giocatore è prigioniero
di un'altra scacchiera
fatta di nere notti e di bianchi giorni

Dio muove il giocatore e questi muove il pezzo.
Che dio dietro a Dio comincia una trama
fatta di polvere e tempo e sogno e agonía?

José Luis Borges - ajedrez (gli scacchi)

4.1 Le simulazioni

Secondo Parisi (2001) le simulazioni sono un nuovo strumento che si è aggiunto agli strumenti tradizionali con cui la scienza cerca di conoscere e di capire la realtà. La rivoluzione scientifica che ha dato l'avvio ai grandi progressi della scienza negli ultimi quattro secoli è stata determinata da due fattori che per la prima volta, nel corso del seicento, si sono trovati insieme: le teorie sono state formulate in termini quantitativi e la prova empirica delle teorie è stata effettuata nelle condizioni controllate e manipolabili del laboratorio sperimentale.

Questo processo ha però riguardato quasi esclusivamente le scienze della natura. Nelle scienze dell'uomo solo saltuariamente si è realizzato un dialogo costante fra teorie e fatti osservati. In questo contesto risulta importante l'apporto che la simulazione può dare alle scienze, in particolare alle scienze dell'uomo.

A che cosa serve la simulazione? Diversi fattori concorrono a formare una risposta:

1. Le simulazioni sono un modo nuovo di esprimere le teorie scientifiche.

Le teorie scientifiche sono i concetti e le idee con cui i ricercatori cercano di individuare i meccanismi, i processi e i fattori che stanno dietro ai fenomeni osservati nella realtà e li spiegano. I concetti e le idee per essere comunicati devono essere espressi in maniera concreta e percepibile.

Fino ad oggi le teorie della scienza sono state espresse mediante simboli, simboli costituiti dalle parole del linguaggio nel caso la teoria venga formulata verbalmente, simboli quantitativi della matematica se la teo-

ria è espressa sotto forma di funzione matematica o simboli costituiti da schemi grafici.

La simulazione, a differenza dei metodi tradizionali usati per esprimere le teorie della scienza, non è espressa sotto forma di simboli, ma come programma di computer. I concetti postulati dalla teoria sono incorporati in un programma, il programma mentre funziona riproduce i fenomeni che la teoria intende spiegare.

Le teorie espresse mediante simboli si limitano a spiegare la realtà, le teorie espresse come simulazioni la riproducono. Anche un programma di computer è fatto di simboli, quelli del linguaggio formale con cui è scritto il programma, quindi anche le simulazioni in quanto teorie espresse sotto forma di programmi sono formulate mediante l'uso di simboli. I simboli usati per esprimere le teorie in maniera tradizionale sono però destinati ad un altro essere umano e da questo devono essere compresi.

I simboli del linguaggio formale con cui è scritto un programma di simulazione non sono destinati ad un essere umano, ma a un computer e sono volti a far sì che il computer svolga certe operazioni.

L'interazione cognitiva tra le teorie-simulazioni e la mente dei ricercatori non passa attraverso la comprensione di questi ultimi dei simboli che costituiscono il programma del computer. Le teorie-simulazioni rivoluzionano così il rapporto a tre fra teorie, predizioni empiriche derivate dalle teorie e mente dello studioso.

I simboli perdono il loro valore semantico poiché non si rivolgono più ad una persona che li deve comprendere, ma ad un calcolatore che li deve interpretare. La teoria è «attiva», non ha più bisogno della mente di uno scienziato per produrre i fenomeni simulati corrispondenti ai

fenomeni osservati nella realtà.

A questo punto uno dei compiti del ricercatore diventa verificare se i fenomeni della realtà e quelli simulati coincidono.

2. Le simulazioni sono laboratori sperimentali virtuali

In un laboratorio sperimentale reale il ricercatore osserva i fenomeni in condizioni controllate, manipola le condizioni che controllano il verificarsi dei fenomeni considerati e osserva le conseguenze, ottenendo così un numero maggiore di informazioni rispetto a quelle di cui disporrebbe se si limitasse a osservare i fenomeni come avvengono al di fuori del laboratorio.

In laboratorio il ricercatore mette inoltre alla prova le predizioni empiriche, le ipotesi tratte da una teoria. Quanto più la teoria dello scienziato spiega le osservazioni effettuate in laboratorio, tanto più la teoria è interessante e apprezzabile.

Una volta costruita, una simulazione diventa un laboratorio sperimentale. Così come accade in un laboratorio reale, i risultati della simulazione si verificano in condizioni che sono controllate dallo studioso. Come in un laboratorio reale il ricercatore può manipolare le condizioni che determinano o influenzano i fenomeni virtuali che costituiscono il risultato della simulazione.

3. Le simulazioni servono per derivare previsioni empiriche dalle teorie

La simulazione automatizza il processo di derivare da una teoria le previsioni da sottoporre a prova empirica. Nella simulazione il compito di derivare previsioni empiriche dalla teoria non ricade più sullo scien-

ziato, ma sul computer.

I risultati stessi della simulazione, una volta che questa è stata effettuata, costituiscono previsioni empiriche.

4. Le simulazioni automatizzano gli esperimenti mentali

Le simulazioni non servono soltanto per scoprire quali predizioni empiriche si possano derivare da una teoria. Esse servono innanzi tutto per elaborare le teorie, per esplorarne e valutarne le caratteristiche e le implicazioni quando sono ancora in fase di costruzione.

Con le simulazioni diventa possibile sviluppare e valorizzare un metodo di ricerca che di solito costituisce appannaggio delle scienze fisiche: il metodo degli esperimenti mentali.

Grazie alle simulazioni è possibile automatizzare gli esperimenti mentali cioè quegli esperimenti non condotti realmente osservando e manipolando la realtà, dove lo scienziato si limita ad immaginare diversi scenari e i risultati a cui si perverrebbe manipolando gli stessi in una determinata maniera.

È difficile utilizzare il metodo degli esperimenti mentali nell'ambito delle scienze dell'uomo. La ragione è che le limitazioni della mente umana riducono molto l'utilità e la fecondità di questo metodo di ricerca. Le cose stanno diversamente quando entrano in gioco le simulazioni. In questo caso, quando uno scienziato formula una ipotesi, la validità della stessa emerge dal confronto con i risultati generati da una simulazione che esprime la teoria di base.

Effettuare un esperimento mentale presenta numerosi vantaggi: è possibile capire meglio la teoria strutturata, si possono ottenere informazioni su quali esperimenti compiere effettivamente, si può utilizzare l'espe-

rienza reale del mondo che ogni ricercatore racchiude nella sua memoria.

Con l'aiuto del computer chiunque può verificare se una predizione tratta da una teoria è avallata dai risultati della simulazione.

5. Le simulazioni sono realtà (artificiali)

La realtà è definita da tre criteri. Essa è a) ciò a cui abbiamo accesso con i nostri sensi, b) quello su cui possiamo agire e che risponde alle nostre azioni e c) ciò che costituisce un vincolo alle nostre azioni ed è nello stesso tempo un mezzo per svolgerle.

La realtà è sia naturale che artificiale. La realtà naturale esiste indipendentemente dall'esistenza degli esseri umani e ha caratteristiche che non sono conseguenza delle azioni degli esseri umani.

Al contrario la realtà artificiale è un prodotto delle azioni degli uomini. Nella realtà artificiale sono compresi la tecnologia, i segnali comunicativi e gli artefatti artistici.

Anche le simulazioni sono realtà artificiale dato che soddisfano i tre requisiti che definiscono la realtà. Una simulazione può essere osservata guardando lo schermo di un computer, è possibile agire su di essa tramite i comandi di un computer e costituisce un vincolo alle nostre azioni poiché con essa è possibile realizzare alcune cose ed altre no.

Per questo motivo costruire una simulazione è creare una realtà artificiale. Una teoria simulazione costituisce una novità per la scienza, in quanto simulazione essa è realtà, le teorie tradizionali invece sono l'esatto opposto della realtà.

Questa novità assimila la scienza, che è volta a conoscere e capire la realtà, alla tecnologia, che serve per modificare e aggiungere aspetti

nuovi alla realtà. Il fatto che le simulazioni non sono solo teorie, ma anche realtà è importante quando si tratta di studiare fenomeni unici come quelli storici che avvengono una volta sola in un dato luogo e in un dato posto.

Le normali teorie, essendo tradizionalmente formulate mediante simboli, sono sempre generalizzazioni che astraggono dai fenomeni individuali, unici, che avvengono in un luogo e un tempo particolari.

Da qui le difficoltà che esse affrontano quando vengono messe a confronto con fenomeni unici. Le teorie simulazioni invece sono realtà. Esse possono riprodurre un evento o un processo unico dato che una realtà può essere una copia di un'altra realtà.

Le simulazioni non nascono dal nulla, esse sono uno sviluppo di idee e di pratiche che già esistevano in precedenza.

Le simulazioni possono essere viste come la fusione di due strumenti già usati dagli scienziati per conoscere e capire la realtà, ma che erano sempre rimasti separati: la costruzione di modelli fisici semplificati della realtà e l'elaborazione di teorie matematiche capaci di cogliere la struttura quantitativa dei fenomeni reali.

Le simulazioni condividono con i modelli fisici il fatto che sono anch'esse un pezzo di realtà costruito per riprodurre e comprendere i fenomeni reali.

Con le formule matematiche le simulazioni condividono il fatto di essere strumenti per automatizzare il processo di derivare previsioni empiriche dalle idee e dalle teorie espresse a proposito dei fenomeni reali.

La novità cruciale che caratterizza le simulazioni è la presenza del computer. Il computer non solo permette di fondere insieme i modelli e le formule matematiche, ma consente di costruire modelli molto più complicati di quelli fisici.

Infine le simulazioni permettono di affrontare molti più fenomeni e aspetti della realtà che non le formule matematiche, non solo perché possono essere anche non quantitative, ma anche perché molti fenomeni reali, pur avendo natura quantitativa, non sono affrontabili matematicamente.

4.2 Vantaggi delle simulazioni

1. Vantaggi delle simulazioni come macchine per derivare predizioni empiriche dalle teorie.

Rendere più semplice il processo di derivazione delle previsioni empiriche dalle teorie presenta una serie di vantaggi per la scienza. Innanzi tutto il compito di derivare predizioni dalle teorie non è più affidato ad un essere umano, ma diventa un processo meccanizzato dentro ad un computer.

Un ricercatore, al momento di effettuare delle previsioni, può essere influenzato dai suoi limiti cognitivi e dalle sue opinioni personali. Con il computer questi problemi si attenuano. Se una simulazione produce determinati risultati questo significa che la teoria espressa nel programma effettivamente implica quei risultati, altrimenti essi non sarebbero stati prodotti. Nel caso che i fatti osservati coincidano con quanto ricostruito nel computer, quanto prodotto dalla simulazione è effettivamente in grado di spiegare la realtà.

Un ulteriore vantaggio che si ha nell'esprimere una teoria nella forma di una simulazione è la garanzia di evitare, nella formulazione della stessa, concetti vaghi, parti mancanti o insufficientemente specificate o dettagliate. Se si cerca di costruire in un computer una teoria-simulazione

che ha questi difetti, il programma non riesce a funzionare o non produce i risultati attesi.

Un terzo vantaggio è collegato con il fatto che le simulazioni sono anche laboratori sperimentali virtuali. Potendo manipolare condizioni e variabili, lo scienziato può ottenere un numero di risultati dalla simulazione molto più grande di quello ottenibile in un laboratorio sperimentale reale. Le simulazioni moltiplicano le predizioni che possono essere tratte da una teoria, aumentandole di numero e aumentando i fattori e le condizioni da cui dipendono, in questo modo esse irrobustiscono l'interfaccia tra teorie e fatti osservati.

2. Vantaggi delle simulazioni come laboratori sperimentali virtuali.

Le simulazioni sono laboratori sperimentali virtuali con potenzialità molto maggiori rispetto a quelli reali. Innanzi tutto nei laboratori virtuali è possibile studiare fenomeni che per ragioni puramente fisiche non è possibile studiare in quelli reali: i fenomeni troppo grandi, quelli che durano troppo a lungo, i fenomeni molto piccoli e quelli che avvengono troppo lontano da noi e che quindi non possono essere manipolati.

Nessuno di questi ostacoli, legati alle caratteristiche della realtà fisica, vale per le simulazioni come laboratori virtuali. Una simulazione può simulare entità o fenomeni grandissimi o piccolissimi, processi che durano molto a lungo o che sono osservabili nella realtà con grande difficoltà.

Un secondo vantaggio delle simulazioni è la possibilità di studiare fenomeni avvenuti nel passato, ma che oggi non sono più presenti. Nel laboratorio reale questi fenomeni non possono essere studiati proprio perché non esistono più.

Un terzo vantaggio è la possibilità di studiare fenomeni complessi senza doverli dividere in sottoparti più semplici e senza doverli isolare dal loro contesto. Quanto detto concorre a mettere fuori gioco il laboratorio sperimentale di tipo tradizionale, ma non mette minimamente in difficoltà quello di tipo virtuale.

Una delle ragioni che spiegano come mai i laboratori reale e virtuale siano soggetti a limiti di differente natura è che nel laboratorio reale la situazione studiata deve essere tenuta insieme dalla mente dello scienziato. È lo scienziato che deve fare attenzione ai diversi aspetti del fenomeno studiato e che deve ricordare quali relazioni esistono tra di loro. È sempre lo scienziato a dover effettuare le previsioni e a sottoporle a prova empirica.

La mente di un essere umano è però soggetta a limiti di memoria, di attenzione e di capacità di ragionamento. Per questo i fenomeni che si prestano ad essere studiati nel laboratorio reale sono quelli che possono essere spiegati da uno o da pochi fattori di cui la mente umana può ricostruire il meccanismo sottostante e la catena di cause ed effetti.

In una simulazione il computer sopperisce ai limiti della mente dell'individuo. Grazie alla memoria e alla capacità di calcolo del computer è possibile immaginare situazioni molto più complesse di quelle che la mente di un essere umano può controllare.

Un ulteriore vantaggio consiste nella possibilità di studiare fenomeni che per ragioni etiche non sono manipolabili. Così risulta impossibile manipolare il sistema nervoso, il corpo, il materiale genetico di un essere umano. Gli esperimenti virtuali si possono invece condurre senza troppi problemi. Infine le simulazioni possono essere più vantaggiose degli esperimenti reali per ragioni puramente pratiche.

Una simulazione richiede relativamente poco tempo, inoltre, per quanto complesso il lavoro di progettazione e scrittura del codice, una simulazione è spesso meno costosa da costruire in termini di tempo e di risorse di un esperimento reale. Infine la materia prima per eseguire esperimenti reali spesso è scarsa o fuori del controllo del ricercatore.

Per esempio è difficile condurre un esperimento psicologico sui disturbi derivanti da una lesione cerebrale. Una volta ricostruito un modello virtuale di sistema nervoso, lo stesso può essere lesionato in modo da ottenere una grande varietà di dati empirici risultanti dalle diverse lesioni simulate.

3. Vantaggi delle simulazioni poiché creano mondi possibili.

Una simulazione ha spesso il compito di spiegare il mondo reale e per farlo lo ricrea. Una differente possibilità è costruire grazie al computer un mondo possibile, cioè un mondo diverso da quello reale, ma con il quale condivide alcuni principi. La scienza può giovare di questi mondi poiché essi possono insegnarle qualcosa sul mondo reale ampliando il campo dei fenomeni con i quali mettere alla prova le teorie.

4. Le simulazioni permettono di studiare i sistemi complessi.

Usare il metodo della simulazione come metodo di ricerca ha delle conseguenze sul modo in cui la scienza concepisce la realtà e si organizza per studiarla.

Una scienza che adotta le simulazioni come strumento di ricerca tenderà a cambiare in due direzioni: tenderà a vedere la realtà come costituita da sistemi complessi e diventerà meno impermeabile alla collaborazio-

ne con altre discipline. La ragione per cui la scienza fino ad oggi si è occupata quasi esclusivamente di sistemi semplici è che questi ultimi si prestano meglio di quelli complessi ad essere analizzati con gli strumenti tradizionali di cui essa dispone.

I sistemi complessi hanno forti componenti di imprevedibilità e irripetibilità che gli strumenti analitici tradizionali hanno difficoltà ad affrontare. In un laboratorio sperimentale lo scienziato manipola una causa e ne osserva l'effetto. Questo processo è appropriato quando si studia un sistema semplice dove una causa determina da sola un dato effetto. In un sistema complesso, al contrario, ogni effetto è il prodotto di molte cause che interagiscono tra di loro al punto che non ha senso cercare di isolare e manipolare una sola di esse. I sistemi complessi inoltre, a differenza di quelli semplici, non possono essere isolati dal contesto in cui operano senza che funzionino in maniera diversa e non possono essere riprodotti in forma identica come richiede il metodo sperimentale.

La realtà è composta in buona parte da sistemi complessi e gli strumenti tradizionali della scienza non bastano per capirne la natura. Le simulazioni sono forse il l'unico strumento atto a studiare la complessità. La teoria alla base di un sistema complesso è anch'essa complessa, riguarda molti elementi, ha molte parti e fa giocare questi elementi e queste parti fra loro con un insieme di interazioni che la capacità di memoria, di attenzione e di ragionamento e i sistemi di comunicazione della mente umana non sono in grado di gestire.

Le cose stanno diversamente se la teoria è espressa come programma di computer, cioè come simulazione. Le risorse cognitive del computer, le sue capacità di memoria, di attenzione, di ragionamento, suppliscono ai limiti delle capacità cognitive umane. Una teoria, per quanto com-

plexa, può essere gestita dal calcolatore.

Gli esseri umani possono inserire gradualmente nel programma le diverse parti della teoria, modificarla, verificarne il funzionamento osservando i risultati e manipolando le variabili della simulazione, ma è comunque il computer che conserva dentro di sé la teoria tutta intera e che la fa girare permettendo all'essere umano di osservarne il comportamento e di capirla.

Il fatto che il metodo sperimentale tradizionale sia adatto per lo studio dei sistemi semplici, mentre per studiare i sistemi complessi siano necessarie le simulazioni dipende da una caratteristica che distingue in maniera fondamentale queste ultime dalla scienza tradizionale: la scienza tradizionale procede attraverso l'analisi dei fenomeni, le simulazioni sono invece un metodo di sintesi dei fenomeni.

La scienza tradizionale scompone la realtà nelle sue componenti e grazie alla teoria e al ragionamento rimette insieme queste componenti per ricostruire i fenomeni. Le simulazioni, invece, seguono la via della sintesi, cioè partono dalle componenti della realtà per scoprire cosa emerge quando le stesse componenti vengono fatte interagire tra di loro. La scienza tradizionale studia e cerca di capire la realtà come la trova, le simulazioni studiano e cercano di capire la realtà ricreandola.

Se la realtà è riprodotta in maniera corretta vuol dire che è stata capita. Adottare le simulazioni significa anche essere spinti verso una visione meno disciplinare della scienza. Le simulazioni, infatti, sono un metodo di ricerca che diversamente da quelli tradizionali si applica ugualmente ad ogni aspetto della realtà. Sono quindi un nuovo linguaggio comune che può essere parlato da qualunque disciplina e attraverso il quale tutte le discipline possono parlarsi.

La visione della realtà come fatta di sistemi complessi spinge nella direzione di una scienza meno disciplinare poiché gli stessi sistemi complessi sembrano avere caratteristiche che sono le stesse quali che siano i fenomeni della realtà che essi rappresentano.

5. Le simulazioni favoriscono l'interdisciplinarietà.

Le simulazioni risolvono molti dei problemi a causa dei quali la scienza fino ad oggi ha dovuto ricorrere ad una sempre più forte suddivisione disciplinare. Consideriamo la grande varietà di fenomeni da cui è composta la realtà, è questa varietà che giustifica la suddivisione della scienza in discipline e sottodiscipline, ciascuna delle quali studia un certo tipo di fenomeni. Nello studio della realtà è spesso necessario studiare assieme fenomeni di natura diversa. I differenti fenomeni sono così affidati a ricercatori appartenenti a discipline diverse, ciascuno dei quali rinuncia a capire i fenomeni stessi nella misura in cui la comprensione non passa attraverso le proprie conoscenze. Le simulazioni cambiano questo stato di cose.

La mente di un ricercatore non può in generale avere familiarità ed esperienza che con un singolo tipo di fenomeni e non riesce a tenere presenti nelle teorie che elabora per spiegare quei fenomeni le interazioni tra i fenomeni stessi ed altri fenomeni di diversa natura. Il computer è soggetto a questo tipo di limiti in maniera inferiore.

Il computer può essere programmato in modo tale che tra i suoi dati figurino informazioni riferite a fenomeni di tipo diverso. Una simulazione può incorporare una teoria che prenda pienamente in considerazione le interazioni tra fenomeni di differente natura. È il computer che conserva le informazioni necessarie, non la mente del ricercatore.

È sempre il computer a calcolare le interazioni tra fenomeni di diversa natura. Per questo le teorie-simulazioni che girano all'interno di un calcolatore tendono a essere teorie più «grandi» rispetto a quelle che stanno nella testa di un ricercatore. Un'altra caratteristica delle discipline e sottodiscipline scientifiche è che non solo si occupano di fenomeni diversi, ma usano metodi di ricerca differenti. Anche da questo punto di vista le simulazioni sono fortemente non disciplinari.

Le simulazioni costituiscono un unico metodo proposto per lo studio di fenomeni di qualunque tipo, che può essere utilizzato in qualunque disciplina e si propongono così come linguaggio unificante della scienza. Le simulazioni hanno natura intrinsecamente non disciplinare in quanto semplificano la realtà.

A differenza degli strumenti tradizionali della scienza sono esse stesse realtà e come tale hanno la tendenza a mantenere l'integrità e la completezza della realtà. Le simulazioni semplificano conservando il tutto in una forma più semplice.

4.3 Problemi delle simulazioni

1. La necessità della verifica esterna

La verifica esterna consiste nello stabilire se le predizioni derivate da una teoria corrispondono ai fatti empirici osservati nella realtà. Chi realizza una simulazione dovrebbe innanzitutto effettuare un confronto tra i risultati ottenuti e i fatti osservati all'interno della realtà empirica, in modo da stabilire un rapporto di corrispondenza.

Uno dei rischi che il ricercatore corre è che il confronto sia troppo in-

tuitivo, non sistematico, parziale. Perché le simulazioni diventino uno strumento sempre più accettato nel campo della ricerca è necessario che la corrispondenza tra i fatti empirici e i risultati della simulazione sia sancita da un confronto esplicito e dettagliato.

Diventa così necessario per il ricercatore possedere una conoscenza della letteratura empirica delle discipline scientifiche. La necessità di acquisire e gestire questo tipo di informazioni può risultare gravoso per chi utilizza il metodo delle simulazioni, che già di per sé richiede parecchio lavoro .

La simulazione, inoltre, ancora non è penetrata a fondo all'interno delle discipline scientifiche tradizionali dove la letteratura empirica viene prodotta. Quando questa penetrazione sarà maggiore, sarà più facile che le simulazioni vengano sottoposte in maniera più rigorosa al vaglio della verifica esterna.

2. Semplificazioni sbagliate

Una delle caratteristiche essenziali che permettono alle simulazioni di individuare i processi e i meccanismi che spiegano i fenomeni empirici è la loro capacità di semplificare la realtà.

Le semplificazioni devono lasciare fuori dalla simulazione gli aspetti irrilevanti dei fenomeni studiati, ma devono includere quelli critici. Un problema che si pone di frequente è che non esiste una regola o un criterio generale per decidere quale semplificazione è giusta e quale è sbagliata.

Diventa ancora più importante l'attenzione con la quale il ricercatore studia l'argomento che sarà oggetto della simulazione.

3. Dettagli arbitrari

Le simulazioni in quanto teorie semplificano la realtà. Ciò non ostante devono costituire sistemi in grado di funzionare all'interno del computer. Si pone così il rischio di introdurre all'interno del modello una serie di dettagli necessari per il funzionamento del sistema nel suo complesso, ma del tutto arbitrari. Diventerà quindi difficile capire in che misura i risultati della simulazione riflettano le caratteristiche del modello più aderenti alla realtà e in che misura riflettano gli aspetti introdotti arbitrariamente.

Il ricercatore avrà, a questo punto, il compito ulteriore di verificare quale sia l'incidenza dei dettagli arbitrari sui risultati totali della simulazione.

4. Confusione tra scopi di conoscenza e scopi pratici

Un ulteriore problema delle simulazioni riguarda il loro rapporto con la tecnologia. Una simulazione può diventare il punto di partenza per la produzione di artefatti tecnologici e per realizzare soluzioni per risolvere problemi pratici. Per questo motivo il confine tra scienza e tecnologia diventa meno netto quando si usano le simulazioni di quando si fa ricorso a metodi più tradizionali.

La tentazione di considerare l'eventuale successo di un'applicazione pratica come la conferma della validità della teoria-simulazione sulla quale essa si basa può costituire un problema. In questo caso è necessario avere sempre ben chiaro che la prova ultima di una simulazione come strumento di indagine scientifica è la corrispondenza tra i risultati della simulazione stessa e i fenomeni empirici osservati nella realtà.

4.4 Le tre grandi separazioni

Secondo Parisi (2001) le scienze dell'uomo sono fondate su tre grandi separazioni che hanno costituito fino ad oggi un grosso ostacolo sulla strada della conoscenza e della comprensione dei fenomeni umani: la separazione tra la mente e la natura, la separazione tra l'individuo e la società e la separazione tra una visione sincronica e una visione diacronica dei fenomeni. Uno dei contributi più importanti che le simulazioni potranno dare alle scienze dell'uomo sarà quello di poter superare queste tre separazioni.

1. La separazione tra mente e natura

La separazione più grande e radicale che la scienza riconosce all'interno della realtà, scrive Parisi (2001), è quella tra la mente e la natura, e quindi tra scienza della mente e scienza della natura. Si tratta di una divisione così antica e incorporata nella cultura che si ha la tendenza a darla per scontata. Essa risale alle origini della tradizione culturale dell'Occidente, che è la tradizione all'interno della quale si è sviluppata la scienza [...]. La separazione tra cultura e mente è una costante nella cultura dell'Occidente ed è un elemento così costitutivo di tale cultura che è stata accettata anche dalla scienza, nonostante che tale separazione rappresenti un'evidente ostacolo all'aspirazione, così intrinseca alla scienza, di costruire un quadro unificato di conoscenza e di comprensione della realtà. Nella scienza la separazione tra mente e natura si manifesta nell'uso, per studiare la mente, di concetti intrinsecamente diversi da quelli usati per studiare la natu-

ra. I fenomeni della natura vengono interpretati dalla scienza come risultanti da processi in cui cause fisico-chimiche producono effetti fisico-chimici e come aventi proprietà in fondo quantitative. Invece una interpretazione del genere non è giudicata possibile o appropriata per i fenomeni della mente. I fenomeni della mente o sono considerati come effetti di cause che non sono fisico-chimiche o non sono considerati affatto effetti di cause, e la loro natura ultima è vista come di tipo qualitativo, non quantitativo.

Le scienze della mente ritengono di sapere quali concetti non sono appropriati per studiare la mente umana, ma hanno difficoltà a chiarire quali siano i concetti adatti.

Le simulazioni si propongono come un nuovo strumento al servizio delle scienze biologiche per arrivare ad una scienza della mente naturalizzata, cioè a una scienza che usa per studiare la mente gli stessi concetti che sono utilizzati all'interno delle scienze naturali. Esistono due ragioni che spiegano l'importanza che le simulazioni possono arrivare a rivestire. Innanzi tutto la mente, cioè il comportamento degli organismi e, nel caso degli esseri umani, la loro vita psichica interiore, è costituita dalle proprietà globali di un sistema complesso di cui fanno parte un numero molto grande di elementi (i neuroni), che producono differenti comportamenti come risultato delle loro innumerevoli interazioni. I metodi tradizionali, sperimentali e matematici delle neuroscienze e delle altre scienze biologiche sono appropriati per studiare i singoli elementi, ma se si vuole studiare le proprietà globali il computer può darci un grosso aiuto.

La seconda ragione per cui le scienze biologiche tradizionali possono

dover ricorrere allo strumento delle simulazioni è che la vita psichica e il comportamento degli organismi, nel caso degli esseri umani, non sono fenomeni che avvengono solo all'interno di un corpo o di un sistema nervoso, ma che riguardano le interazioni tra l'organismo e l'ambiente. Il comportamento non può essere studiato restando dentro al corpo o al sistema nervoso (come fanno le scienze biologiche), ma analizzando le interazioni complesse tra organismo e ambiente.

Tra gli strumenti simulativi che si fondano su di una radicale abolizione di ogni separazione concettuale tra natura e mente un posto particolare merita la Vita Artificiale.

La Vita artificiale, scrive Parisi (2001), simula organismi dotati di un corpo avente determinate dimensioni e una determinata forma. Il corpo al suo interno contiene un sistema nervoso e degli organi che interagiscono tra di loro e con il corpo. All'interno della Vita Artificiale è simulato anche l'ambiente in cui l'organismo vive. L'ambiente può modificarsi nel tempo anche come risultato del comportamento degli elementi presenti al suo interno.

2. La separazione tra individuo e società

La seconda grande separazione che interessa la scienza è quella che si osserva tra l'individuo e la società. Sempre secondo Parisi (2001) sia gli individui che la società sono fenomeni reali che la scienza ci deve aiutare a comprendere. Questi due fenomeni si condizionano reciprocamente: senza gli individui le strutture e i fenomeni sociali non esisterebbero; d'altro canto i prodotti collettivi del comportamento umano influenzano in maniera profonda gli individui. Ciononostante le scienze dell'uomo hanno sempre avuto grande difficoltà a considerare insieme

i due fenomeni. Le divisioni disciplinari hanno assegnato ad alcune scienze (psicologia e scienza cognitiva) lo studio degli individui, del loro comportamento e della loro mente, e ad altre scienze (quelle sociali) lo studio dei prodotti collettivi del comportamento degli individui. Questa divisione di compiti ha impedito che le interazioni e le influenze tra i due livelli di fenomeni potessero essere studiate nel modo appropriato. La psicologia e la scienza cognitiva ignorano le strutture sociali e culturali, le scienze sociali ignorano la mente dell'individuo e, quando cercano di tenerne conto, applicano modelli del comportamento individuale non corrispondenti alla realtà.

La scienza tiene separati individui e società poiché gli strumenti tradizionali che ha a disposizione non le consentono di studiare insieme i due fenomeni. Le scienze che studiano gli individui fanno uso del metodo sperimentale, metodo presente in maniera molto meno diffusa all'interno delle scienze sociali.

Un'altra causa della divisione tra scienze dell'individuo e scienze della società riguarda la maniera di formulare le teorie. Fino a che queste sono espresse con i metodi tradizionali, cioè verbalmente o con formule matematiche, diventa difficile dare conto insieme dei fenomeni individuali, di quelli sociali e delle loro interazioni.

Le simulazioni rendono possibile alle scienze dell'uomo di studiare insieme gli individui, le società e il modo in cui essi interagiscono e si influenzano fra di loro permettendo di gestire teorie molto complesse.

Le simulazioni che aboliscono la separazione tra individui e società si chiamano simulazioni ad agenti e cercano di riprodurre i fenomeni sociali in quanto emergenti dal comportamento di individui che interagiscono tra loro. Gli individui appartenenti ad una simulazione di

questo tipo sono capaci di rispondere in determinati modi agli stimoli e alle situazioni all'interno delle quali essi si trovano di volta in volta. Gli agenti interagiscono tra loro rispondendo a stimoli forniti da altri agenti e assumendo comportamenti che serviranno a loro volta da stimolo per altri individui.

Il risultato di queste interazioni è l'emergere di quei fenomeni collettivi studiati dalle scienze sociali. Naturalmente, come accade in tutti i sistemi complessi, i fenomeni sociali causati dal comportamento degli agenti non sono predicibili sulla base di come essi si comportano individualmente. Tra gli effetti del comportamento degli agenti, sostiene Parisi (2001), vi è il comparire di strutture e istituzioni sociali che costituiscono l'ambiente in cui essi si muovono e che di conseguenza influenzano i loro comportamenti.

Come deve essere il modello simulativo dell'agente? Per rispondere a questa domanda dobbiamo chiederci se vogliamo dare per scontata la tradizionale divisione tra mente e natura oppure se vogliamo superare questa distinzione e adottare un punto di vista naturalistico nel definire cosa è un agente.

Un agente può essere modellato in modo non naturalistico usando i tradizionali concetti mentalistici della psicologia oppure i più formalizzati concetti della scienza cognitiva derivanti dall'analogia tra mente e computer. Ad esempio un agente può essere modellato come un insieme di regole, formulate in modo simbolico, cioè mediante parole ed eventualmente simboli logici e matematici, che specificano qual è il comportamento dell'agente a seconda delle diverse cir-

costanze. Non c'è alcun riferimento al fatto che l'agente è un organismo fisico, che il suo comportamento è guidato da un sistema nervoso, che ha un corpo e un materiale genetico ereditato frutto di una storia di evoluzione biologica. Nelle simulazioni che usano un modello non naturalistico degli agenti, gli agenti tendono ad essere tutti uguali [...]. Simulazioni di questo tipo permettono di superare la separazione tra individui e società, ma non quella tra natura e mente [...]. Esiste invece un secondo tipo di simulazioni sociali ad agenti che cercano di superare nello stesso momento, cioè all'interno di una stessa simulazione, tutte e due le separazioni. In questo tipo di simulazione gli individui sono simulati per quello che sono, cioè organismi dotati di un corpo, di un sistema nervoso che guida il loro comportamento, di un materiale genetico ereditato, che vivono in un ambiente fisico e hanno interazioni fisiche con tale ambiente. Si tratta delle simulazioni della Vita Artificiale [...]. L'unica differenza è che ora la dimensione sociale dell'ambiente viene in primo piano. Parisi (2001).

3. La separazione tra sincronia e diacronia

La terza grande separazione sulla quale si basano le divisioni disciplinari nelle scienze dell'uomo è quella tra una visione dei fenomeni umani che considera i fenomeni stessi come indipendenti dal loro passato e come se esistessero fuori dal tempo e una visione che li considera, invece, come prodotti del cambiamento nel tempo degli esseri umani e della loro società.

La prima visione, che chiamiamo sincronica, è alla base di tutte le scienze dell'uomo fatta eccezione per quelle storiche. La seconda, che chiamiamo diacronica, è quella delle scienze storiche. La separazione tra lo studio sincronico e quello diacronico degli esseri umani costituisce un altro serio ostacolo per la comprensione dei fenomeni umani. Secondo Parisi (2001), l'approccio sincronico è, entro certi limiti, giustificato per quanto riguarda le scienze della natura. Gran parte dei fenomeni studiati dalla fisica, dalla chimica e dalla biologia sono fenomeni che non cambiano nel tempo e possono perciò essere studiati senza guardare al passato. Questa è una delle ragioni per cui l'esperimento di laboratorio ha un ruolo così fondamentale nelle scienze della natura. Nel laboratorio il ricercatore studia fenomeni che avvengono e che sono manipolati in quel momento.

Le cose non stanno così per quanto riguarda le scienze dell'uomo. La maggior parte dei fenomeni che riguardano gli esseri umani sono oggi differenti da quello che erano in passato. I fenomeni attuali non sono «speciali» rispetto a quelli passati, anzi, da questi dipendono direttamente poiché ne sono l'evoluzione e senza di questi non possono essere capiti. Per comprendere a fondo questi fenomeni è necessario adottare una «epistemologia genetica¹», cioè bisogna ricostruire e capire il percorso che ha portato i fenomeni ad evolversi fino a diventare quello che sono ora.

Le simulazioni tendono ad abolire la separazione tra sincronia e diacronia. La ragione pratica fondamentale che giustifica la divisione tra scienze sincroniche e diacroniche è la difficoltà di studiare la realtà nel suo complesso, sia al momento attuale che nel passato, senza

¹La definizione è dello psicologo Jean Piaget

suddividerla in discipline.

Come in altri casi, le simulazioni sono antidisciplinari, in quanto riducono il peso di questa ragione pratica, dice Parisi (2001). È impossibile pensare ad una scienza dell'uomo unica, che sia nello stesso tempo sincronica e diacronica, se tutto quello che si ha a disposizione sono i libri e le teste degli scienziati. Le cose cambiano se si ha a disposizione un computer. Il computer può contenere molti più dati quelli che possono essere contenuti in un singolo libro o nella testa di uno scienziato, e permette di collegare e far interagire tra loro i dati contenuti nei libri di una intera biblioteca o nelle teste di una intera comunità di scienziati. Ma soprattutto il computer permette di formulare teorie di ogni tipo di società, attuale o del passato, e della storia che ha condotto a quella società, teorie le cui predizioni sono esplicite e osservabili da chiunque.

Le teorie espresse come simulazione, inoltre, rendono chiare quali sono le predizioni che si ricavano da una teoria che vuole spiegare fatti ed eventi del passato. Così quanto si conosce del passato diventa evidenza empirica a favore o sfavore della teoria formulata. Secondo Parisi (2001) una storia che formuli teorie esplicite a proposito della società umana del passato è spinta a integrarsi con le scienze sociali sincroniche (la sociologia, l'antropologia culturale, l'economia, la scienza politica...). Le scienze sociali sincroniche sono piene di teorie e di concetti a cui la storia può attingere per costruire le sue simulazioni sul passato delle società umane e dei cambiamenti che queste hanno sperimentato.

D'altro canto quello che manca alle scienze sociali sincroniche è una più stretta interazione tra le tante teorie ed elaborazioni concettuali che le caratterizzano con i dati empirici. In questo caso è la storia che può venire incontro alle scienze sociali grazie alla sua descrizione dei dati empirici. Le simulazioni diventano il luogo di incontro tra scienze sociali sincroniche e diacroniche. L'unione di prospettiva sincronica e diacronica resa possibile dalle simulazioni diventa un modo per superare la separazione tra dati empirici e teorie che costituisce una delle principali cause di arretratezza delle scienze dell'uomo.

Con le simulazioni scompare ogni differenza tra simulare una società attuale o del passato. L'integrazione che risulta tra sincronia e diacronia ha il vantaggio di rendere ogni studio della società uno studio genetico, cioè uno studio che cerca di capire la natura di un fenomeno attraverso la ricostruzione del suo passato.

4.5 Simulazione e agenti

Secondo Ostrom (1988), i ricercatori che operano nel campo delle scienze sociali possono avvalersi di tre differenti sistemi simbolici per descrivere la realtà: i sistemi verbali, quelli matematici e la simulazione tramite computer. Nel caso delle rappresentazioni verbali, sostengono Gilbert e Terna (2000), il problema per lo studioso è comprendere a pieno le conseguenze delle idee formulate. I modelli matematici, ancorché più solidi di quelli verbali, hanno lo svantaggio di rappresentare i fenomeni sociali in maniera troppo complicata per essere maneggiabili analiticamente senza introdurre delle semplificazioni che si rivelano spesso sbagliate e fuorvianti.

La diffusione esponenziale delle simulazioni ad agenti osservata negli ultimi

anni può essere interpretata come il tentativo di trovare una strada alternativa per lo studio dei fenomeni sociali complessi. In questa luce, osserva Terna (1998), è importante che i ricercatori affinino le loro conoscenze informatiche per riuscire a sfruttare le possibilità offerte dal proliferare degli studi nel campo delle scienze sociali che si avvalgono di modelli informatici ad agenti. Un aiuto ai ricercatori, continua Terna (1998), può venire dall'uso di ambienti di sviluppo per le simulazioni ad agenti, che permettono la realizzazione di simulazioni senza imporre un onere di studio dell'informatica troppo pesante. I sistemi multi agenti nascono all'interno di una branca dell'intelligenza artificiale chiamata intelligenza artificiale distribuita. Lo scopo dell'intelligenza artificiale distribuita era quello di affrontare i problemi proposti suddividendoli tra differenti agenti, ognuno dotato di diverse capacità. Anche se inizialmente questo strumento fu usato per trovare soluzioni a problemi concreti, scrivono Gilbert e Terna (2000), fu presto realizzato che esso sarebbe stato utile anche per studiare i fenomeni sociali, facendo corrispondere un agente ad ogni individuo o organizzazione della realtà.

Dal punto di vista dei sistemi multi agente, un agente è dotato di autonomia (controlla le proprie azioni), socievolezza (interagisce con gli altri agenti tramite un qualche «linguaggio»), reattività (percepisce l'ambiente in cui vive e risponde agli stimoli che questo gli offre) e determinazione (è in grado di compiere delle azioni atte a raggiungere uno scopo predeterminato).

Sostiene Axtell (2000) che derivano diversi vantaggi dall'uso delle simulazioni ad agenti rispetto all'uso dei modelli matematici. Innanzitutto è molto facile limitare la razionalità degli agenti o renderli gli uni diversi dagli altri, in modo da rendere la simulazione più realistica rispetto ai modelli dove operano agenti tutti uguali e perfettamente razionali. Inoltre, all'interno di molti processi sociali esistono elementi di grande importanza, come lo spazio fisico, che

sono difficili da includere in un modello matematico. Purtroppo, continua Axtell (2000), esiste anche uno svantaggio: Per potersi rendere conto della robustezza delle teorie formulate sotto forma di simulazione non basta far funzionare il modello una volta sola, ma è necessario fare vari tentativi cambiando ogni volta le condizioni e i parametri iniziali per poter stabilire una relazione di causa ed effetto tra gli elementi del modello e i risultati ottenuti. Sempre Axtell (2000) presenta tre possibili usi delle simulazioni ad agenti:

- Nel primo caso immaginiamo che sia possibile descrivere matematicamente un fenomeno sociale, inoltre il modello può essere risolto analiticamente o numericamente. Se il modello può essere risolto analiticamente, la simulazione ad agenti è solamente uno strumento per presentare i risultati, se può essere risolto numericamente essa rappresenta una analisi di Monte Carlo.
- Nel secondo caso è possibile scrivere le equazioni che descrivono il fenomeno studiato, ma non è possibile risolverle. In queste circostanze è frequente introdurre delle asserzioni nel modello matematico per riuscire a pervenire ad una soluzione. Purtroppo queste asserzioni si rivelano spesso poco realistiche. In questo caso la simulazione ad agenti può servire ad illustrare le proprietà dinamiche del modello e per verificare quale relazione sussiste tra le asserzioni e i risultati.
- Nell'ultimo caso abbiamo le situazioni in cui il modello matematico non può essere neppure scritto. La simulazione ad agenti può rivelarsi allora l'unico strumento di indagine che abbiamo a disposizione per studiare sistematicamente i fenomeni sociali.

Axtell (2000) aggiunge che esistono due ragioni supplementari che rendono la simulazione ad agenti un argomento di vivo interesse all'interno dello stu-

dio delle scienze sociali. Innanzitutto lo sviluppo tecnologico dei computer permetterà entro breve di gestire simulazioni dove interagiranno un numero impressionante di agenti. In secondo luogo, continua Axtell (2000), ci troviamo in un momento critico per quanto riguarda lo sviluppo dei programmi di simulazione ad agenti. Assistiamo ad un utilizzo sempre più vasto di queste metodologie, ma alcune questioni fondamentali rimangono senza risposta: un sistema dove gli agenti sono distribuiti e decentralizzati è più o meno utile di un sistema dove gli agenti hanno un controllo centrale maggiore? Come devono essere strutturati gli agenti perché il sistema di cui fanno parte funzioni nel modo migliore?

Capitolo 5

jES.

5.1 A cosa serve jES.

Il programma jES (Java enterprise simulator) nasce allo scopo di studiare le dinamiche che portano alla creazione di un'impresa, ai suoi possibili miglioramenti e all'interazione dell'azienda stessa con il mondo esterno.

Secondo Gibbons (2000), il modello economico di base di un'impresa è stato per molto tempo un oggetto misterioso, una sorta di scatola nera in cui si immettevano i fattori produttivi e dalla quale si otteneva l'utile d'esercizio. Clarkson e Simon (1960), dello stesso avviso, criticano le modalità con le quali è stato a lungo condotto lo studio dell'impresa.

Con grande ingenuità assumiamo il ricavo come grandezza fondamentale e rappresentiamo i rischi sotto forma di distribuzioni di probabilità. Fatto questo rimaniamo nell'incertezza. Abbiamo costruito una teoria che spiega come l' homo oeconomicus assume una decisione o una teoria che spiega come egli deciderebbe se potesse ragionare solo in termini di numeri, non possedendo concetti di tipo qualitativo o verbale, Clarkson e Simon (1960).

Quando all'interno del processo di decisione si hanno soltanto variabili non numeriche i limiti dell'analisi classica emergono chiaramente.

Simon in particolare, partì dallo studio dell'organizzazione aziendale per passare poi a studiare il comportamento umano in generale e ipotizzò l'esistenza di una razionalità limitata che è propria dell'uomo, non soltanto nei processi decisionali di tipo economico, ma per quelli riguardanti tutti i campi delle scienze sociali.

La scelta della simulazione ad agenti ci dà la possibilità di studiare situazioni caratterizzate dalla razionalità limitata dei soggetti decisionali e dalla difficoltà di tradurre la realtà in termini matematici.

Secondo Axtell (2000) possiamo distinguere tre ambiti nei quali usare la simulazione. Nel primo caso, il problema studiato può essere descritto interamente con un sistema di equazioni risolvibili analiticamente (nel qual caso il modello ad agenti si riduce ad un mezzo grazie al quale presentare i risultati) o numericamente.

Nel secondo caso, il problema può essere descritto matematicamente, ma non può essere risolto interamente. In questo caso il modello ad agenti può servire per capire, tramite esempi empirici, quale sia la relazione esistente tra i risultati ottenuti e le teorie postulate.

Nell'ultimo caso, il problema non può essere descritto con una formula e la simulazione ad agenti resta l'unica possibilità per affrontare il problema.

Grazie a jES, possiamo descrivere un'impresa in termini di ricette (sequenze di passi produttivi) e di unità produttive in grado di eseguirle. Questa descrizione nasce dall'interazione degli agenti della simulazione all'interno di un ambiente specifico di cui conosciamo le caratteristiche. A differenza di altri strumenti di simulazione, jES, sviluppato a partire dal codice del programma Swarm, utilizza sia agenti non dotati di arbitrarietà (le cose da fare

e le unità che sanno eseguirle) che agenti capaci di ‘decidere’.

Le possibili applicazioni di jES sono tre:

1. L’ottimizzazione delle organizzazioni tramite analisi dei differenti scenari derivanti dall’introduzione di cambiamenti all’interno della realtà simulata.

In questo tipo di simulazione si procede alla creazione di un modello della realtà al fine di capire come funziona l’organizzazione. Una volta che il modello riproduce fedelmente i fatti realmente accaduti si introducono i diversi cambiamenti presi in considerazione e si studiano le conseguenze che questi hanno sullo stesso.

Vi è una probabilità ragionevolmente grande che dette conseguenze si verificherebbero anche se i cambiamenti simulati venissero effettivamente introdotti nella realtà.

2. L’interazione tra il modello e gli esseri umani rappresentati da agenti artificiali.

Nella simulazione capita sovente di dover effettuare una scelta. Abbiamo una serie di criteri in base ai quali essa può essere compiuta: possiamo decidere a caso (per quanto possa sembrare inverosimile, questo criterio fornisce spesso una buona approssimazione di ciò che accade effettivamente). Possiamo utilizzare regole di scelta fisse. Possiamo creare un sistema esperto, al quale appoggiarci nel momento della decisione. Possiamo chiedere al computer di scegliere per noi, con l’ausilio di strumenti di ottimizzazione come gli algoritmi genetici o i classifier system. Possiamo, infine, affidarci alla scelta di un essere umano.

L’ultima possibilità offre il vantaggio di poter studiare come le persone si comportano all’interno di una organizzazione e di poter formare nuovo personale, lasciando che questo, interagendo con il simulatore,

impari a valutare le conseguenze delle proprie azioni. Per poter interagire con la simulazione, le persone hanno bisogno di un agente artificiale che ne sia 'l'incarnazione' all'interno del simulatore. Questi agenti, a cui diamo il nome di *avatar*, possono ricevere ordini dagli individui che rappresentano, dialogando grazie ad un programma apposito.

3. L'analisi teorica delle dinamiche che portano alla nascita, allo sviluppo e al declino di un'organizzazione.

Consideriamo tali dinamiche da due differenti prospettive. Nel primo caso l'impresa è giudicata in base al profitto percepito dai suoi dipendenti. Il successo dell'impresa si riflette positivamente sulla ricchezza dei lavoratori, ma una crescita esagerata dell'azienda farà sì che il contributo di ogni singolo lavoratore al successo aziendale sia minimo, disincentivando così il personale dall'impegno sul lavoro. Secondo Axtell (1999), infatti, le imprese di successo sono quelle che sanno attirare e trattenere dipendenti motivati.

Nel secondo caso le imprese sono considerate come il luogo dove operano le idee e le scelte dell'imprenditore. Questo concetto si presta ad una doppia interpretazione: Secondo Kirzner (1997) l'impresa è soggetta a meccanismi di prova ed errore, dove gli errori compiuti costituiscono un'opportunità di guadagno che funge da incentivo per le azioni correttive che si rendono necessarie. Per Burt (1992), siamo in presenza di un modello dinamico, nel quale l'imprenditore individua e sfrutta gli spazi liberi tra le zone già caratterizzate da intensi rapporti d'affari.

È utile sottolineare, infine, l'importanza che rivestono l'auto organizzazione e la complessità nel processo di simulazione. La possibilità che le imprese si adattino alle nuove condizioni che si presentano, sulla base del comportamento di agenti adattivi, è per Epstein(2003), la caratte-

ristica principale che una simulazione dovrebbe possedere, sia essa del primo o del secondo tipo.

5.2 Come funziona jES?

Per capire come funziona jES è necessario rendersi conto dell'esistenza di tre differenti formalismi che operano all'interno delle simulazioni che intendiamo realizzare: il WD (what to do), il DW (which is doing what) e da ultimo il WDW (When doing what). La realtà, all'interno della simulazione, sarà rappresentata tramite una serie di ricette (sequenze di passi che devono essere eseguiti per produrre un bene o concludere compito).

le ricette contengono l'aspetto WD della simulazione, si incaricano cioè di descrivere ciò che deve essere eseguito. All'interno della simulazione troviamo anche le unità produttive, unità che sono in grado di realizzare le ricette e che costituiscono l'aspetto DW della stessa.

Il terzo formalismo (WDW) si riferisce alla sequenza temporale secondo la quale si verificano gli eventi. L'idea di base sulla quale si regge jES è che i formalismi WD e DW sono separati, esattamente come nel mondo reale lo sono le unità produttive e i prodotti da queste realizzati.

Possiamo osservare come quasi tutta l'intelligenza del processo di simulazione è situata nell'aspetto WD. Il processo dal punto di vista del codice di programmazione si comporta come segue:

- le unità produttive si caratterizzano per la presenza di una lista d'attesa nella quale vengono registrati gli ordini in attesa di lavorazione e sulla base della quale esse agiscono.

Il processo di lavorazione è diviso in un'unità temporale (tic). Al termine di ogni tic l'unità di produzione farà riferimento alla lista d'attesa perché le sia assegnata una nuova lavorazione.

Gli ordini, una volta ultimata una fase di lavorazione (step), vengono inseriti in un'apposita lista (made production list) che sarà successiva-

mente trasmessa alle altre unità produttive. Se un'operazione richiede più di un tic per essere eseguita, l'ordine di lavorazione rimane all'interno dell'unità produttiva e le viene riassegnato immediatamente e in maniera diretta;

- durante il processo produttivo vengono lanciati nuovi ordini (e nuove ricette in essi contenute). Notiamo che ogni ordine contiene una ricetta composta di passi produttivi (step) che devono essere portati a compimento. I nuovi ordini entrano a far parte della simulazione
 - seguendo un ordine temporale scandito da un listato che costituisce il formalismo WDW oppure
 - in maniera casuale grazie allo OrderGenerator [vedi infra].

I nuovi ordini saranno assegnati alle unità produttive secondo il procedimento descritto al punto seguente.

- Ogni ordine custodito nelle made production list delle unità produttive effettua un'indagine all'interno del mondo simulato alla ricerca di unità produttive che possano completare la fase successiva della sua lavorazione.

Se una sola unità produttiva si rende disponibile alla lavorazione, l'ordine le sarà assegnato automaticamente. Nel caso in cui più di una unità produttiva sia in grado di portare a termine il compito, la lavorazione potrà essere assegnata

- all'unità che figura per prima nel file unitData/unitBasicData.txt; che elenca tutte le unità che entrano a far parte della simulazione,
- a una qualsiasi delle unità scelta a caso;
- all'unità con la lista d'attesa più corta.

La possibilità di effettuare una scelta per quanto riguarda il criterio di assegnamento degli ordini alle unità produttive si rivela particolarmente interessante in quanto introduce la possibilità che in futuro jES si presti a un lavoro di interazione con esseri umani a fini didattici (per allenare i dipendenti di un'ipotetica organizzazione a prendere decisioni e valutare in seguito tali decisioni) o a fini di ricerca (per comprendere in che modo le persone assumono le decisioni).

Infine esiste la possibilità di introdurre in futuro un criterio di scelta basato su strumenti di ottimizzazione come gli algoritmi genetici o i classifier system. Quando l'ultima fase produttiva viene ultimata, l'ordine viene abbandonato dopo essere stato contabilizzato;

- se il parametro useNewses è acceso, ogni unità produttiva avverte le altre unità produttive nel caso un ordine sia in via di ultimazione;
- il processo ricomincia continuamente da capo man mano che sia avvicinando le unità temporali.

Per quanto riguarda i tic è utile far notare come la sincronicità e il parallelismo temporali sono ottenuti nel modo seguente: ogni unità produttiva svolge in maniera indipendente nello stesso tic i compiti sopra descritti relativi ad ogni singolo punto, prima di passare al punto successivo.

Basic recipe

The basic recipe in an order is structured as follows:

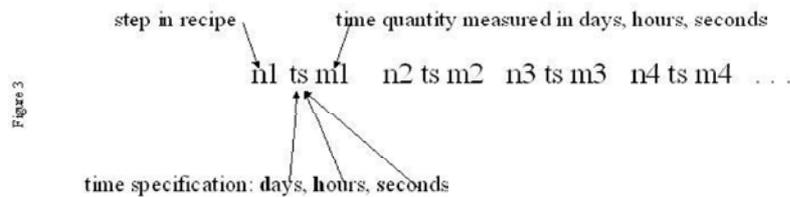


Figura 5.1: Lo schema di una ricetta

5.3 L'aspetto What to Do

Guardando più da vicino l'aspetto WD, scopriamo accanto agli ordini (composti da ricette) e alle ricette (insiemi di differenti step che vengono eseguiti per una determinata durata temporale) un ulteriore concetto, quello di layer (strato).

Due ordini contenenti la stessa ricetta possono differenziarsi da un punto di vista qualitativo. Il concetto di layer introduce appunto differenze di tipo qualitativo tra due ordini. Il numero di layer che possiamo utilizzare viene determinato grazie al parametro `totalLayerNumber`. L'attribuzione di un ordine a un differente layer è effettuata dall'utente del programma in sede di scrittura della ricetta.

Vi sono casi in cui non sarebbe realistico immaginare processi produttivi che

Sequential batch

a *sequential batch* in a recipe repeats for m time units the production of the same step of the recipes of b orders (the unit doing step n_3 has obviously to hang around until it has b similar order in its waiting list to start the batch; while waiting, it can be working on other batches or single orders)

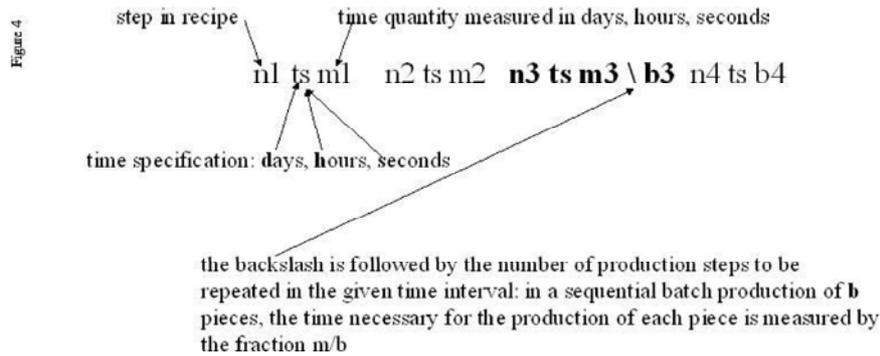


Figura 5.2: Un sequential batch

abbiano a che fare con oggetti singoli. È utile per ciò introdurre il concetto di batch (catasta). I batch possono essere di due tipi: i sequential batch e gli stand alone batch. Nel caso dei sequential batch, per compiere alcune fasi della lavorazione, è necessario che questa possa essere applicata ad un insieme di oggetti, ad un batch appunto. Ciò non toglie che durante le altre fasi della lavorazione gli oggetti vengano lavorati singolarmente.

Uno stand alone batch è molto simile ad un sequential batch, ma è incluso in una ricetta a sé di cui costituisce l'unica fase di lavorazione. La ricetta in questione descrive un processo a sé stante (ad esempio un ordine di approvvigionamento che un'impresa effettua di tanto in tanto).

jES possiede anche capacità computazionali, che possono essere utilizzate durante le fasi di lavorazione di un ordine. L'uso delle capacità computazionali permette a jES di eseguire previsioni, simulare aste per la scelta

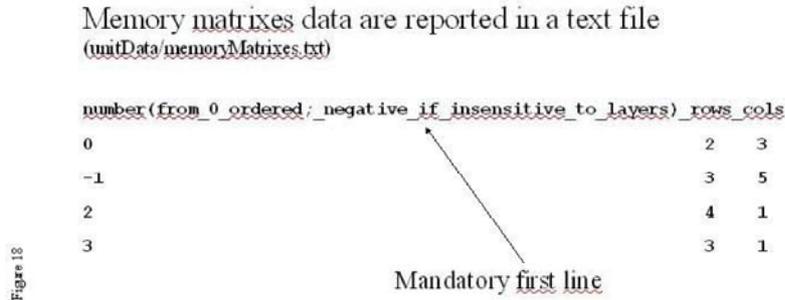


Figura 5.3: unitData/memoryMatrixes.txt.

degli approvvigionamenti o nel caso del 118 di far dialogare le ambulanze con la centrale operativa di Grugliasco. I calcoli sfruttano i dati contenuti in matrici chiamate MemoryMatrix che sono create sul modello del file unitData/memoryMatrixes.txt. La figura seguente è relativa al file in questione.

Le memoryMatrix utilizzano i layer in maniera automatica, per impedirlo basta far precedere il numero identificativo della memoryMatrix da un segno di negatività. Seguono alcuni esempi di ricette che contengono passi computazionali.

L'external format si riferisce al formato in cui le ricette vengono scritte dall'utente, l'intermediate format, ancora comprensibile da parte dell'utente, è frutto di una prima traduzione che il computer esegue in maniera automatica e che può essere facilmente interpretato grazie ai commenti presenti nel file

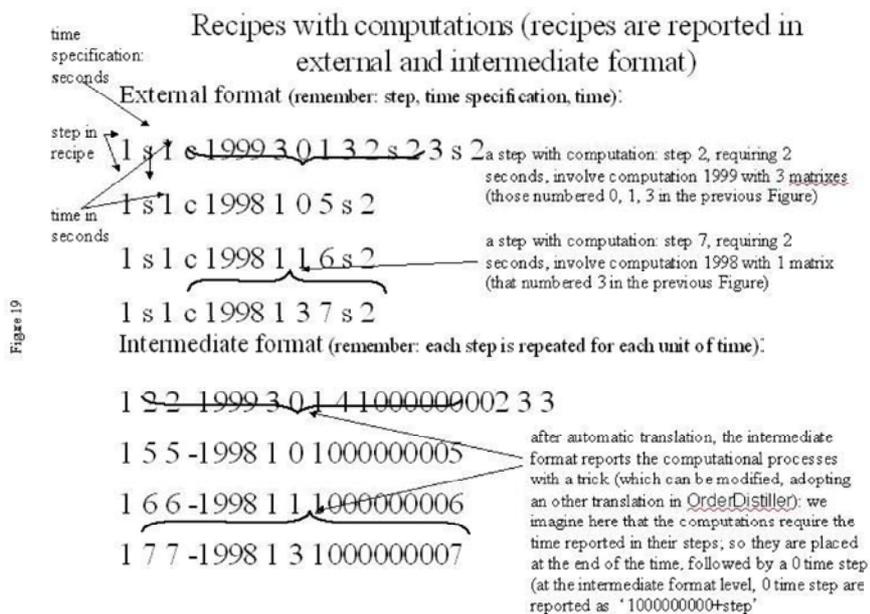


Figura 5.4: Ricette che sfruttano passi computazionali.

OrderGeneretor.java.

Esiste un terzo formato, l'internal format, che può essere esaminato solo tramite la lettura del file Order.java.

Il passo computazionale funziona nel modo seguente: si pone una c alla quale fa seguito il numero identificativo del passo computazionale che deve essere effettuato. Seguono il numero di matrici che devono essere utilizzate e i numeri identificativi di ogni singola matrice usata. A questo punto il codice Java presente nel file ComputationalAssembler.java interagisce con le ricette sopra riportate. Nelle due figure successive viene fornito un esempio relativo al codice java.

In un ordine contenente la ricetta '1 s 1 c 1998 1 0 5 s 2', alla fine dei due tic (unità temporali) necessari per eseguire la fase 5, la matrice 0 effettua un'operazione di scrittura in posizione (0,0). Nel codice java presente nella

The Java Swam code used by the recipes with computation of this example

Figure 20

```
/** computational operations with code -1998 (a code for the checking
 * phase of the program
 *
 * this computational code place a number in position 0,0 of the
 * unique received matrix and set the status to done
 */
public void c1998(){

    mm0=(MemoryMatrix) pendingComputationalSpecificationSet.
        getMemoryMatrixAddress(0);
    layer=pendingComputationalSpecificationSet.
        getOrderLayer();

    mm0.setValue(layer,0,0,1.0);
    mm0.print();

    done=true;
} // end c 1998
```

Figura 5.5: Il codice java.

figura si può vedere come questa operazione che interessa la matrice $mm=0$ si collega alla matrice effettiva grazie al metodo `getMemoryMatrixAdress`, il metodo `setValue` si occupa invece di inserire il valore voluto nella posizione $(0,0)$. Alla fine il passo computazionale viene decretato concluso.

Un ulteriore esempio è fornito dalla figura seguente.

In questo caso, consideriamo l'ordine contenente la ricetta '1 s 1 c 1999 3 0 1 3 2 s 2 3 s 2'. Alla fine delle due unità temporali necessarie per effettuare la fase 2 le matrici 0, 1 e 3 subiscono un'operazione di controllo volta a verificare se le posizioni $(0,0)$ siano vuote in corrispondenza del layer d'appartenenza. In caso la posizioni siano occupate, c 1999 si occupa di svuotarle e di decretare successivamente concluso il passo computazionale.

Nel codice riportato in figura, le matrici $mm=0$, $mm=1$ e $mm=2$ sono collegate alle matrici effettive grazie al metodo sopra descritto. L'effetto delle

The Java Swam code used by the recipes with
computation of this example

Figure 21

```
/** computational operations with code -1999 (a code for the checking
 * phase of the program
 *
 * this computational code verifies position 0,0 of the three
 * received matrices, only if these positions are all not empty
 * the code empties them and set the status to done
 */
public void c1999() {
    mm0=(MemoryMatrix) pendingComputationalSpecificationSet.
        getMemoryMatrixAddress(0);
    mm1=(MemoryMatrix) pendingComputationalSpecificationSet.
        getMemoryMatrixAddress(1);
    mm2=(MemoryMatrix) pendingComputationalSpecificationSet.
        getMemoryMatrixAddress(2);
    layer=pendingComputationalSpecificationSet.
        getOrderLayer();

    if( (mm0.getEmpty(layer,0,0) || mm1.getEmpty(layer,0,0)
        || mm2.getEmpty(layer,0,0) ) )
    {
        mm0.setEmpty(layer,0,0);
        mm1.setEmpty(layer,0,0);
        mm2.setEmpty(layer,0,0);
        done=true;
    }
} // end c1999
```

Figura 5.6: Il codice java bis.

quattro ricette presentate nella figura 19 è il seguente: la ricetta contenente il passo c 1999 non può completare la fase 2 senza che si siano dapprima prodotti gli effetti di una delle ricette contenenti il passo c 1998.

Quando infine la ricetta contenente il passo c 1999 potrà ultimare la fase 2 di lavorazione e procedere alla fase successiva, gli effetti delle ricette contenenti il passo c 1998 saranno eliminati.

unit_#	usewarehouse	prod.phase_#	fixed_costs	variable_costs
2	0	2	0	0
101	0	100	0	0
102	0	100	0	0
103	0	100	0	0
104	0	100	0	0
201	0	200	0	0
202	0	200	0	0
203	0	200	0	0
204	0	200	0	0
205	0	200	0	0
206	0	200	0	0
207	0	200	0	0
208	0	200	0	0
209	0	200	0	0
303	0	303	0	0
302	0	302	0	0
71	0	0	0	0
55	0	0	0	0
65	0	0	0	0
90	0	0	0	0
100	0	0	0	0
110	0	0	0	0
115	0	0	0	0
125	0	0	0	0
135	0	0	0	0
140	0	0	0	0
150	0	0	0	0
160	0	0	0	0
210	0	0	0	0
230	0	0	0	0
240	0	0	0	0
245	0	0	0	0
250	0	0	0	0
260	0	0	0	0
270	0	0	0	0
280	0	0	0	0
285	0	0	0	0
290	0	0	0	0
300	0	0	0	0
301	0	0	0	0
310	0	0	0	0
320	0	0	0	0
330	0	0	0	0

Figura 5.7: unitData/unitBasicData.txt.

5.4 L'aspetto which is Doing What

Per quanto riguarda l'aspetto DW è necessario a questo punto differenziare le unità produttive semplici (simple production units) da quelle complesse (complex production units) e introdurre il concetto di end units. Per unità produttiva si intende la più piccola struttura capace di eseguire una o più lavorazioni. Le unità produttive semplici sono in grado di svolgere un solo tipo di lavoro e vengono descritte nel file unitData/unitBasicData.txt

Il file raccoglie le seguenti informazioni: i numeri che identificano ogni

inventory. In caso contrario, l'ultimo elemento sarà costituito da uno 0. All'interno del file unitData/unit.xls sarà presente un foglio di calcolo per ogni unità complessa.

Da ultimo, per end unit si intende la fase finale di una lavorazione quando questa non si conclude nel modo sopra descritto. In questo caso all'interno della ricetta relativa al processo produttivo sarà presente un segno 'e' seguito dal numero identificativo di un'unità che non avrà caratteristiche produttive, ma che rappresenterà un magazzino dove sarà riposto l'oggetto risultante dal processo produttivo appena conclusosi. Le end unit sono descritte nel file unitData/endUnitList.txt cui fa riferimento la figura seguente.

I codici numerici presenti in questo file, fanno riferimento alle end unit e possono presentare o meno un iniziale segno di negatività a seconda che esse siano caratterizzate o meno dalla presenza di layer.

```
end_unit_#;_use_positive_code_for_layer_sensitive_end_unit;_negative_for_
r_unsensitive;_do_not_duplicate_the_codes,_neither_with_a_different_sig
n
1
2
3
4
5
6
7
11
14
15
16
17
18
19
20
21
22
23
24
25
26
1100
1101
1102
1103
1104
1105
1106
1107
1108
1109
1110
1111
11001
11002
11003
11004
500
501
502
503
```

Figura 5.9: unitData/endUnitList.txt.

5.5 OrderDistiller e orderGenerator

Il meccanismo di simulazione è attivato dagli ordini contenenti le ricette che a loro volta elencano le fasi di lavorazione e i tempi necessari per portarle a compimento. Gli ordini possono alternativamente essere

- generati a caso dall'orderGenerator (esemplare della classe OrderGenerator), nel caso ci interessi mettere alla prova il codice da noi scritto o nel caso in cui non disponiamo di informazioni riguardo alla sequenza temporale degli ordini presenti nella nostra realtà simulata;
- estratte dall'orderDistiller (esemplare della classe OrderDistiller) da un elenco di ricette, seguendo una determinata pianificazione temporale.

Nel primo caso le ricette sono generate internamente dal programma e vengono messe in gioco a intervalli costanti. Nel secondo caso viene introdotto l'aspetto WDW che costituisce il terzo formalismo della simulazione e sarà trattato nel paragrafo successivo.

5.6 Il formalismo When Doing What

Il formalismo WDW è costituito da un insieme di convenzioni utili per creare un repertorio di ricette e per disporle all'interno dello spazio temporale. Il suddetto insieme di ricette è contenuto nel file `recipeData/recipes.xls`. All'interno di questo foglio di calcolo i dati vengono letti in maniera sequenziale dall'alto verso il basso e da sinistra verso destra. Sono ammesse le spaziature. Ogni ricetta comincia con il proprio nome e con un numero identificativo, di seguito troviamo il corpo della ricetta scritto secondo le regole fin qui analizzate. La ricetta si conclude con un segno ';'. Tutti gli elementi della ricetta sono scritti uno per cella.

La sequenza con la quale vengono lanciati gli ordini è contenuta nel file `recipeData/orderSequence.xls`. In questo file ogni riga di codice descrive gli avvenimenti che intercorrono in un determinato momento della giornata (ad un determinato tic). Ogni riga fa riferimento ad un tic e comincia con un numero identificativo, seguito dal codice della ricetta che deve essere lanciata in quel determinato momento. Seguono un segno di moltiplicazione e un fattore di ripetizione che indica quante volte la ricetta sarà lanciata.

Nella stessa riga possono essere lanciate di seguito ricette che fanno riferimento a numeri identificativi diversi. Al termine di ogni riga è necessario apporre un segno ';', anche se la riga si presenta vuota. Una volta arrivato alla fine, il file `recipeData/orderSequence.xls` è eseguito nuovamente dal principio in maniera automatica.

Il file `recipeData/orderStartingSequence.xls` contiene esattamente le stesse informazioni del file `recipeData/orderSequence.xls`, ma viene utilizzato solo una volta all'inizio della simulazione, per esempio per assicurarsi che vi sia immediata disponibilità degli approvvigionamenti richiesti, nel caso la simulazione coinvolga un'impresa (nel caso del 118 il file `recipeDa-`

ta/orderStartingSequence.xls è l'unico che viene utilizzato, non verificandosi gli interventi di assistenza in maniera ciclica).

5.7 La contabilità

jES è in grado di contabilizzare in maniera automatica le attività delle unità produttive e gli ordini eseguiti. Vi è un doppio sistema di contabilizzazione:

- ad ogni ordine vengono attribuiti costi fissi e variabili in relazione al numero di unità produttive utilizzate all'interno del processo di produzione;
- le unità produttive che non entrano nella lavorazione non hanno costi variabili e i loro costi fissi non vengono attribuiti a nessun ordine di produzione. È utile notare che se un'unità produce un inventario, questo è soggetto ad un costo finanziario, misurato con un tasso d'interesse.

Va aggiunto che jES contabilizza anche i costi relativi all'acquisto esterno di beni che entreranno in gioco una volta lanciato un ordine.

Anche i ricavi sono contabilizzati. Essi comprendono gli ordini ormai ultimati moltiplicati per il loro prezzo di vendita, il valore degli inventari in un momento determinato e il valore degli ordini semilavorati in un momento specifico.

I benefici dell'impresa simulata sono calcolati nel modo seguente: ai ricavi totali viene sommato il valore delle scorte delle varie unità e il valore dei semilavorati. Dal risultato sono poi sottratti i costi totali e quelli finanziari.

5.8 I parametri della simulazione

I parametri della simulazione possono essere modificati in due modi diversi: o all'inizio della simulazione stessa tramite i pannelli che compaiono sullo schermo una volta avviato jES o modificando il file `jesframe.scm` e vengono elencati nell'ordine che segue:

- `displayFrequency`, che si incarica di determinare ogni quanti tic i grafici relativi alla simulazione vengono aggiornati;
- `verboseChoice`, se attivo, si incarica della produzione di commenti relativi alle attività interne al programma;
- `printMatrixes`, se attivo, mostra su schermo il contenuto delle matrici relative ai passi computazionali;
- `checkMemorySize`, attiva messaggi relativi alla memoria usata;
- `unitHistogramXPos` e
- `unitHistogramYPos`, coordinate dell'angolo superiore sinistro della finestra contenente l'istogramma relativo alle unità produttive;
- `endUnitHistogramXPos` e
- `endUnitHistogramYPos`, coordinate della finestra contenente l'istogramma relativo alle end unit;
- `timeToFinish`, momento durante il quale la simulazione viene interrotta;
- `ticksInATimeUnit`, definisce quanti tic dell'orologio interno compongono un'unità temporale;

- totalUnitNumber, si riferisce al numero di unità semplici o complesse che esistono all'interno della simulazione;
- totalEndUnitNumber, si riferisce al numero di endUnits esistenti all'interno della simulazione;
- totalLayerNumber, numero di layer che possiamo usare;
- totalMemoryMatrixnumber, numero totale delle memoryMatrix;
- sameStepLifoAssignment, serve per rendere continua una lavorazione da parte di una medesima unità;
- assignEqualStepToSameUnit, serve per assegnare alla stessa unità produttiva la stessa fase produttiva appartenente a due processi diversi;
- compareDisregardingUnits, se attivo, è possibile comporre un batch partendo da oggetti prodotti da differenti unità;
- maxTickInAUnit, se acceso, gli ordini che restano in attesa di lavorazione oltre ad un certo numero di tic sono eliminati;
- useWarehouses, serve per permettere alle unità di accumulare scorte;
- useNewses, serve per rendere attiva la trasmissione di informazioni tra unità;
- maxinWarehouses, definisce la dimensione delle scorte al di sotto della quale l'inventory production è permessa;
- minInWareHouses, dimensione minima delle scorte;
- infDeepness, si riferisce a quanto le unità produttive possono guardare in avanti in termini di tempo quando trasmettono e ricevono informazioni;

- `inventoryFinancialRate`, è relativo al costo finanziario delle operazioni di `inventory`;
- `inventoryEvaluationCriterion`, è relativo alla modalità con cui si applica il `financial rate`;
- `revenuePerEachRecipeStep` e
- `revenuePerCostUnit`, criteri per valutare i ricavi: se il primo è acceso il secondo è spento e viceversa;
- `nOfNewsesToProduce`, numero di news da produrre compreso tra il numero minimo e quello massimo delle scorte esistenti nelle warehouse;
- `nOfNewsesToBeCleared`, numero delle news che devono essere eliminate a seguito di una decisione che prevede un aumento di produzione per incrementare le scorte;
- `nOfOrdersInNewses`, numero degli ordini per i quali vengono trasmesse informazioni;
- `orCriterion`, possono esistere all'interno della simulazione casi in cui sia data una scelta tra due possibili azioni. A seconda del valore presente in `orCriterion`, il programma si comporterà diversamente;
- `orMemoryMatrix`, interviene in caso il valore posto nell'`orCriterion` implichi un processo di scelta tra due possibili azioni che ricorre all'uso di un passo computazionale;
- `unitCriterion`, nel caso più di una unità produttiva sia in grado di eseguire una lavorazione, a seconda del valore dello `unitCriterion` (0,1 o 2), la lavorazione sarà assegnata alla prima unità libera come figura

dal file `unitData/unitBasicData.txt`, ad un'unità libera presa a caso o all'unità libera dotata della lista d'attesa più corta;

- `noAccountingInFirstTimeUnit`, se attivo, per tutta la durata della prima unità temporale non si effettuerà alcun processo di contabilizzazione;
- `useOrderDistiller`, determina l'uso dell'`orderDistiller` se acceso, in caso contrario verrà utilizzato l'`orderGenerator`;
- `maxStepNumber`, numero massimo di fasi contenute all'interno di una ricetta prodotta internamente via `orderGenerator`;
- `maxStepLength`, numero massimo di unità temporali che possono essere attribuite ad un singola fase della lavorazione in una ricetta prodotta internamente via `orderGenerator`.

Capitolo 6

Il servizio ‘118’

6.1 Il 118.

Con il Decreto del Presidente della Repubblica del 27 marzo 1992 , il 118 si configura come l’unico numero di riferimento per quanto riguarda il pronto intervento sanitario, aggiungendosi ad un vasto panorama fino ad allora formato esclusivamente da associazioni locali e volontarie¹.

Con il DPR il 118 è investito della facoltà di cancellare d’ufficio tutti gli altri numeri che si riferiscono ad associazioni di pronto intervento, dirottando l’intero traffico telefonico sulla propria linea. Questo potere è esplicitamente illustrato nell’articolo tre comma tre della legge, il quale cita:

‘L’attivazione della centrale operativa comporta il superamento degli altri numeri di emergenza sanitaria di enti, associazioni e servizi delle unità sanitarie locali nell’ambito territoriale di riferimento, anche mediante convogliamento automatico delle chiamate sulla centrale operativa del 118.’

¹Questo capitolo è stato scritto in comune con Fabiana Guerra.

In realtà, in Piemonte questo non è avvenuto ed il 118 si è aggiunto ad un vasto e variegato panorama formato da numeri di pronto intervento di associazioni locali e volontarie.

Durante questi anni sono stati adottati principalmente due strutture organizzative, la prima che descriviamo è quella che gli addetti ai lavori chiamano ‘gestione verticale’.

Questo modello organizzativo è adottato in presenza di modesti volumi di traffico (intesi come numero di richieste di assistenza). Il funzionamento di questo tipo di struttura è piuttosto semplice: la chiamata arriva direttamente ad un infermiere che, dopo aver interrogato il paziente per poter effettuare una valutazione del caso e assegnarvi un codice di urgenza, si vede costretto ad interrompere il contatto per allertare un mezzo di soccorso.

È chiaro come in questo tipo di organizzazione non vi sia se non una minima possibilità di effettuare interventi di primo soccorso telefonico che in alcuni casi potrebbero risultare vitali.

La seconda possibile struttura organizzativa è quella denominata a ‘gestione orizzontale’ e costituisce il modello organizzativo adottato della centrale operativa di Grugliasco, dove il traffico si aggira attorno alle 600.000 telefonate annue.

La gestione della centrale operativa di Grugliasco avviene come segue: una chiamata arriva al POF (Posto Operatore Filtro) il quale ha come unico compito quello di smistare le chiamate destinate alla guardia medica (che si trova fisicamente all’interno della struttura della centrale operativa anche se si tratta di una organizzazione distinta ed autonoma rispetto a questa) da quelle indirizzate al 118. Queste ultime sono inoltrate alle PVS (postazioni di valutazione sanitaria).

Per effettuare queste operazioni gli operatori assegnati al POF devono pos-

sedere competenze meramente tecniche.

Dopo essere passata dal POF la chiamata arriva alle PVS nelle quali si effettua una valutazione dello stato del paziente e dove si compila una scheda che viene inoltrata al box ambulanze.

Gli infermieri che si trovano alla postazione di valutazione possono prestare i primi interventi, cercando di far eseguire alcune semplici operazioni di soccorso ai presenti sul luogo dell'incidente. Gli interventi a distanza in alcuni casi possono risultare vitali. Gli operatori che si occupano di questa fase del pronto intervento devono necessariamente essere infermieri professionali.

Le chiamate sono ordinate dalle PVS in base a un criterio di urgenza. Da qui le chiamate sono trasmesse al box ambulanze, nel quale si trovano due operatori tecnici, uno assegnato alla provincia l'altro alla città di Torino, che gestiscono le comunicazioni tra ambulanza e centrale operativa.

Ad affiancare questi operatori si trova un medico che interviene nel caso in cui vadano prese importanti decisioni sanitarie quali, per esempio, la scelta dell'ospedale, o l'invio di un altro mezzo nel caso vi sia stata una sottostima della gravità dell'intervento.

Dopo la scelta del mezzo incomincia la 'missione operativa', che si compone di otto passi:

- Ricerca mezzo: l'operatore si avvale dell'aiuto del computer il quale è in grado di fornire un elenco dei mezzi disponibili.
- Allarme al mezzo: l'ambulanza viene allertata dall'operatore, che preoccupa anche di fornire i codici relativi alla missione. Questi codici sono composti da numeri che indicano la criticità dell'intervento, la natura della patologia e da lettere che si riferiscono al luogo dove si deve effettuare il soccorso.

- Partenza: il mezzo parte alla volta del luogo da dove è stata richiesta assistenza. Comincia la routine di soccorso. Durante questa fase il contatto radio con la centrale operativa avviene in maniera periodica. A questa fase assegnamo il nome di ‘monitoraggio’.

Il monitoraggio si compone delle comunicazioni che avvengono tra centrale operativa e mezzo di soccorso. Queste comunicazioni sono le seguenti:

- Sul posto: il mezzo segnala che si trova sul luogo dell’incidente.
- Paziente a bordo, in partenza: il mezzo segnala che il paziente si trova a bordo e che si sta partendo alla volta dell’ospedale più idoneo.
- Arrivo all’ospedale.
- Di nuovo operativi: qui finisce la fase di monitoraggio ed il mezzo, dopo le ordinarie operazioni di pulizia e di riattrezzaggio, si rende nuovamente disponibile per nuove missioni.
- Chiusura servizio: il mezzo contatta la centrale operativa per aggiornare definitivamente la banca dati per quanto riguarda il paziente e quindi per chiudere definitivamente il caso.

Il 118 dispone di tre tipi di mezzi di soccorso: i MSA (mezzi di soccorso avanzato) sul quale sono presenti 1 medico, 1 infermiere e 2 soccorritori volontari, i MSAB (mezzi di soccorso avanzato di base) sul quale sono presenti 1 infermiere e 2 soccorritori volontari e i MSB (mezzi di soccorso di base) sul quale sono presenti 2 soccorritori volontari.

Nell’area urbana operano 5 MSA, 1 MSAB e 9 MSB. Nella provincia operano 14 MSA, 9 MSAB e 76 MSB.

I mezzi appartengono ad associazioni di volontariato (es. Croce Rossa, Croce Verde ecc.) e vengono messi a disposizione del 118 in base a due tipi di

convenzioni. La prima è chiamata h24 e prevede che i mezzi siano garantiti a disposizione 24 ore su 24. La seconda è una convenzione estemporanea: la copertura non è garantita sulle 24 ore e sui sette giorni la settimana.

È chiaro come queste diverse coperture incidano molto sul servizio di pronto intervento, infatti può capitare che l'ambulanza più vicina non sia disponibile perché è un mezzo in estemporanea.

Alcuni fattori intervengono a complicare il modello: innanzitutto la gestione comune delle chiamate di soccorso ricade interamente sul POF, che oltre a provvedere all'inoltro delle chiamate ai PSV deve fungere da centralino per le altre associazioni (in particolar modo per la guardia medica), evidenziando così un primo collo di bottiglia nel modello.

Un secondo aspetto da considerare riguarda il tempo previsto di 90 secondi per la gestione chiamate dal POF al box ambulanze.

I 90 secondi previsti si riferiscono ad un singolo evento. Il sorgere di svariati eventi nello stesso momento può rappresentare un serio problema, possono infatti formarsi delle code.

Lo stesso problema può verificarsi per la routine di soccorso.

Un'ulteriore complicazione che riguarda i casi di trauma (i quali secondo i responsabili del servizio rappresentano il 20% degli interventi), in questi frangenti la centrale operativa è costretta a allertare vigili del fuoco e forze dell'ordine.

Si deve ovviamente ricordare che i mezzi che non possiedono copertura temporale continua potrebbero non essere disponibili e che devono quindi essere depennati dalla lista dei possibili mezzi d'intervento.

Dopo questo breve excursus sul modo di operare della centrale operativa proviamo a schematizzare quanto detto con il seguente grafico:

6.2 I dati del modello

Una volta ricostruito sulla carta il modello della struttura organizzativa idel 118, è diventato fondamentale procurarsi un ampio numero di dati relativi agli interventi effettuati.

I dati ci sono stati forniti dalla centrale operativa di Grugliasco.

Purtroppo le informazioni raccolte si sono rivelate carenti sotto svariati punti di vista:

- non erano presenti informazioni relative ai tempi operativi delle POF (postazioni operatori filtro);
- le informazioni disponibili non operavano una distinzione tra i tempi operativi delle PVS (postazioni di valutazione sanitaria) e quelli dei Box ambulanze;
- le informazioni erano spesso incomplete (spesso erano disponibili due soli dati: l'ora in cui la centrale operativa riceveva una richiesta d'aiuto e l'ora in cui si concludeva la missione operativa.
L'assenza di dati intermedi era indice della presenza di carichi di lavoro elevati all'interno della centrale operativa. In queste condizioni, gli operatori non avevano tempo materiale di registrare tutti i dati relativi alle operazioni di soccorso;
- Le postazioni di chiusura davano per terminate svariate missioni di soccorso nello stesso momento (sempre per risparmiare tempo);
- gli indirizzi relativi agli utenti del servizio di soccorso, degli ospedali minori e delle associazioni di volontariato proprietarie degli autoveicoli erano spesso insufficientemente dettagliate;

- non era segnalato dove stazionassero abitualmente le ambulanze;
- non erano presenti dati riguardanti i tempi di rifornimento di carburante o di ‘riattrezzaggio’ (pulizia e rifornimento medicinali) delle ambulanze.

Per ovviare a questi inconvenienti si è deciso di concentrare inizialmente gli sforzi sugli interventi effettuati in area urbana, tralasciando per il momento la provincia di Torino (gli interventi relativi alla provincia sono stati inseriti nella simulazione iniziale, ma con un dettaglio minore rispetto agli interventi effettuati all’interno della città di Torino).

Questa decisione rispondeva a due esigenze: quella di concentrare le forze in un ambito limitato (avremmo potuto scoprire che la simulazione non afferrava la realtà dei fatti, nel qual caso sarebbe stato necessario ricominciare da capo) e quella di poter utilizzare inizialmente i dati migliori (relativi appunto alla città di Torino).

Per riuscire ad utilizzare comunque la maggior parte dei dati in nostro possesso ci siamo comportati come segue:

- abbiamo assegnato alle POF un tempo standard, per farlo abbiamo chiesto il parere del dott. Ghiselli, direttore della centrale operativa di Grugliasco;
- abbiamo operato nello stesso modo per assegnare un tempo standard ai Box ambulanze.
Per sottrazione abbiamo trovato i tempi operativi delle PVS;
- per i primi periodi ci siamo ‘accontentati’ di dividere il tempo operativo totale relativo ai singoli interventi per il numero di operazioni di cui questi si componevano.

Più avanti abbiamo effettuato un rigoroso lavoro di pulizia dei dati [vedi infra];

- gli indirizzi di ospedali, utenti e associazioni di volontariato non sono serviti per portare a termine la simulazione iniziale.

In un secondo momento abbiamo cercato di ricavare quante più informazioni possibili tramite ricerche effettuate con l'internet e domande rivolte ai responsabili della centrale operativa di Grugliasco;

- abbiamo scoperto che all'interno della città di Torino le ambulanze stazionano presso un determinato ospedale e che tendono a farvi ritorno a meno che ricevano disposizioni in senso contrario;
- inizialmente non abbiamo avuto bisogno dei dati relativi ai tempi di 'riattrezzaggio' e di rifornimento carburante.

A questo punto lo scopo del nostro lavoro è diventato ricostruire 'nel computer' una giornata del 118.

A tale scopo abbiamo scelto volutamente una giornata che presentasse un numero medio di missioni di soccorso, onde evitare che la comprensione delle dinamiche interne del servizio di pronto soccorso fosse distorta dal verificarsi di eventi straordinari dovuti al numero eccessivo o non sufficiente di interventi effettuati.

Come risultato della simulazione si sarebbero potuti presentare due diversi scenari:

- i risultati della simulazione avrebbero potuto non coincidere con quanto effettivamente verificatosi durante la giornata che volevamo rappresentare. Questo risultato sarebbe stato imputabile a diversi possibili fattori:

1. Innanzitutto il modello avrebbe potuto non corrispondere alla struttura organizzativa del 118 poiché gli addetti ai lavori (impiegati e responsabili) avevano dei dubbi a proposito di come funzionasse la struttura in cui lavoravano. In questo caso la simulazione, una volta corretta, avrebbe permesso agli addetti ai lavori di ‘capirsi meglio’.
 2. Il modello avrebbe potuto non corrispondere a causa di un errore non concettuale avvenuto durante la sua costruzione;
- i risultati della simulazione avrebbero potuto coincidere con quanto effettivamente accaduto nella realtà.

Questo sarebbe servito come prova implicita della bontà della simulazione.

Di fatto i risultati ottenuti contribuirono a comporre uno scenario misto.

I risultati erano abbastanza buoni da farci scartare l’idea che il modello fosse stato costruito in maniera errata. La presenza di alcuni inconvenienti, tuttavia, ci impediva di considerare il modello perfetto.

Nella fattispecie gli inconvenienti erano costituiti dall’esistenza di code di attesa non giustificate che indicavano che alcune unità produttive all’interno del 118 (POF, PVS, ambulanze...) oltre a svolgere i compiti loro assegnati al momento, avrebbero dovuto, potendo, svolgerne altri. Questi compiti formavano una coda e aspettavano che una determinata unità si liberasse per portarli a termine.

In una simulazione che si propone di riprodurre la realtà dei fatti (simulazione ex-post) questo risultato non è coerente.

Durante la giornata presa in esame, il 118 aveva prestato soccorso ad un numero determinato di persone, le ambulanze erano andate e venute dagli

ospedali ai luoghi degli incidenti, i POF e i PSV avevano svolto il loro lavoro, tutto questo era effettivamente accaduto.

I dati riguardavano i fatti avvenuti, non aveva senso che comparissero delle code.

La spiegazione di questo strano risultato non poteva essere che una: i dati che ci erano stati forniti e che noi avevamo aggiustato per quanto possibile non ci permettevano di evitare alcuni errori.

Per esempio l'abitudine di chiudere più interventi nello stesso momento faceva sì che alcune ambulanze non ancora libere fossero chiamate nuovamente ad operare, con la conseguenza nascita di una coda di attesa.

Si rendevano necessari alcuni interventi volti a migliorare i dati posseduti da una parte e ad aumentare la verosimiglianza del modello dall'altra.

6.3 Uso di passi computazionali per lanciare ricette.

Durante la costruzione del modello ci si è resi conto di alcune eccessive semplificazioni².

In primo luogo la comunicazione tra le ambulanze ed il box, e viceversa, non era considerata all'interno delle ricette, il che portava a nascondere un collo di bottiglia che nella realtà esiste e che è una seria preoccupazione dei dirigenti della centrale operativa.

Nello specifico il problema è rappresentato dal fatto che le ambulanze ed il box, durante la loro operatività, sarebbero tenuti a comunicare sei volte tra loro:

- nel momento in cui il box allerta le ambulanze;
- quando queste partono per raggiungere il luogo dell'incidente;
- quando arrivano sul luogo dell'incidente;
- quando ripartono verso l'ospedale;
- quando raggiungono l'ospedale;
- quando si danno nuovamente operative e quindi pronte a effettuare un altro intervento.

In realtà i dati forniti risultano fortemente incompleti poiché questa procedura raramente viene eseguita fedelmente.

Il problema della comunicazione potrebbe verificarsi anche nell'operazione

²Questo capitolo è stato scritto in comune con Carlo Desotgiu e Fabiana Guerra a partire da un lavoro eseguito da Carlo Desotgiu.

di chiusura del servizio. Avviene infatti, in seguito all'arrivo all'ospedale dell'ambulanza e all'accettazione del paziente al pronto soccorso, una comunicazione tra gli operatori che hanno effettuato il servizio e la centrale operativa, riguardo ai dati della persona soccorsa.

Questa fase avviene in contemporanea con le operazioni necessarie per rendere l'ambulanza nuovamente operativa senza interromperle per effettuarla. Essendo questi dei problemi reali e gravosi per il corretto servizio di pronto intervento sanitario si è deciso di sviluppare una procedura adeguata, ponendo particolare attenzione alla prima delle due perché è sicuramente quella che crea maggiori disagi nel caso in cui dovessero esserci molte richieste d'aiuto e quindi molte ambulanze da gestire.

In prima analisi la soluzione più facilmente praticabile sembrava essere quella di dedicare delle unità di tempo (tic dell'orologio della simulazione) all'intervento del box durante l'operatività delle ambulanze, al fine di rappresentare la comunicazione tra le due unità.

E' immediatamente emerso che l'idea, seppur plausibile, peccava di realismo in quanto significava che ad ogni comunicazione l'ambulanza si sarebbe fermata per un unità temporale parlando con il box, ripartendo poi al momento successivo. Ovviamente nella realtà questo non avviene e le operazioni si svolgono in contemporanea, il che ha reso necessaria un'altra soluzione.

Successivamente si è pensato di sviluppare il formalismo AND, (attualmente non ancora attivo in jES) per le sue caratteristiche di contemporaneità. Nella figura sottostante viene rappresentato un esempio del funzionamento di questo meccanismo: nella ricetta si possono avere percorsi paralleli che devono essere eseguiti simultaneamente in un dato momento.

In particolare, nella ricetta sottostante la fase n1 viene eseguita per m1 tic e,

Parallel paths in recipes

after each step we can have also parallel paths in our recipes, all to be simultaneously executed at a specific time

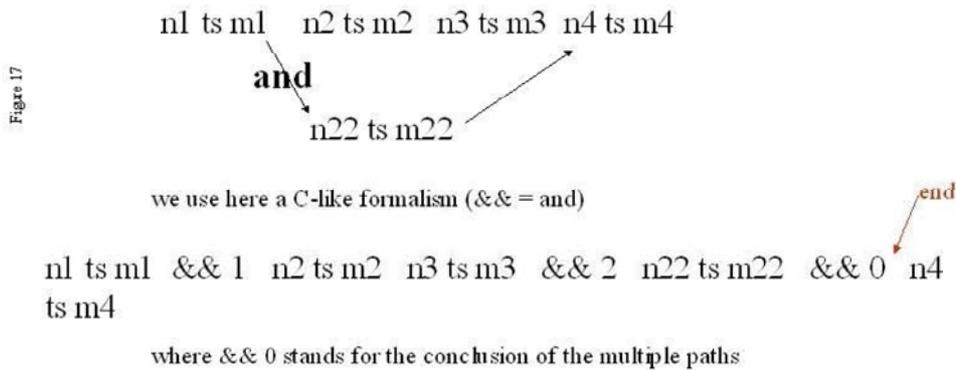


Figura 6.2: Rami paralleli nelle ricette.

in contemporanea, vengono eseguite le seguenti fasi così inserite nella ricetta:

`&& 1 n2 ts m2 n3 ts m3 && 2 n22 ts m22 && 0 n4 ts m4.`

La spiegazione è che il branch 1 (indicato con `&& 1`) esegue le fasi `n2` e `n3`, in contemporanea il branch 2 (indicato con `&& 2`) esegue la fase `n22`.

Quando entrambi rami sono conclusi viene eseguita la fase `n4`.

`&&0` indica la conclusione del processo AND.

Dalla spiegazione si capisce il motivo per cui si è ricorsi ad una procedura differente.

Si ha infatti la necessità che i processi siano paralleli, ma che la fase successiva proceda indipendentemente dalla conclusione di entrambi.

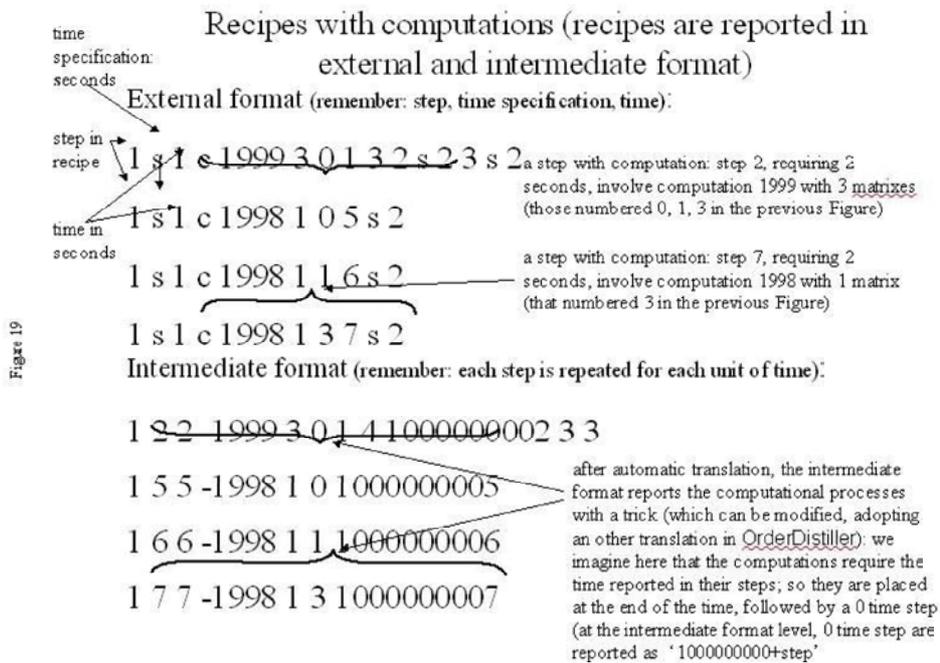


Figura 6.3: Ricette con passi computazionali.

Nel nostro caso questo implica che l'ambulanza, dopo aver comunicato con il box, prosegue il servizio indipendentemente dall'avvenuta ricezione o meno da parte del box. Lo stesso discorso si applica al caso inverso.

La conclusione a cui si è giunti consiste nell'eseguire, all'interno della ricetta principale, un passo computazionale, il quale attiva una sottoricetta che esegue la comunicazione; le due ricette proseguono poi indipendentemente.

La figura in alto riporta un esempio di come il formalismo viene inserito nelle ricette.

La fase 1 è eseguita per un tic (unità temporale), dopodiché la fase 2 richiama il passo computazionale numero 1999 (indicato con 'c 1999') il quale indica quante e quali matrici devono essere usate.

In questo esempio il numero di matrici da utilizzare è 3 e le matrici da usare

The Java Swarm code used by the recipes with the
example code c 1999

```
/** computational operations with code -1999 (a code for the checking
 * phase of the program
 *
 * this computational code verifies position 0,0 of the three
 * received matrixes; only if these positions are all not empty
 * the code empties them and set the status to done
 */
public void c1999(){
    mm0=(MemoryMatrix) pendingComputationalSpecificationSet.
        getMemoryMatrixAddress(0);
    mm1=(MemoryMatrix) pendingComputationalSpecificationSet.
        getMemoryMatrixAddress(1);
    mm2=(MemoryMatrix) pendingComputationalSpecificationSet.
        getMemoryMatrixAddress(2);
    layer=pendingComputationalSpecificationSet.
        getOrderLayer();

    if( ! (mm0.getEmpty(layer,0,0) || mm1.getEmpty(layer,0,0)
        || mm2.getEmpty(layer,0,0) ) )
    {
        mm0.setEmpty(layer,0,0);
        mm1.setEmpty(layer,0,0);
        mm2.setEmpty(layer,0,0);
        done=true;
    }
} // end c1999
```

Figura 6.4: Il codice c 1999.

sono identificate con i numeri 0, 1, 3; queste contengono le procedure che devono essere eseguite. Sotto riportiamo il codice corrispondente al passo computazionale dell'esempio.

Nel nostro caso il passo computazionale attiverà la ricetta indicata all'interno delle matrici integrando nell'OrderDistiller i dati dell'OrderStartingSequence e dell'OrderSequence.

In un caso come il nostro, dove per il momento le ricette sono esclusivamente di tipo ex-post, le modifiche apportate ad OrderDistiller è sufficiente che

operino in `OrderStartingSequence`.

Come risultato si avrà l'aggiunta in coda alla riga dell'`OrderStartingSequence`, processata al momento, di un secondo evento, che fa riferimento alla ricetta di cui sopra.

I due eventi sono simultanei, in quanto, l'`OrderDistiller` lancia le ricette e poi le lascia al loro destino. `jES` opera in modo distribuito con ogni ricetta che procede, come gli avvenimenti nella realtà, indipendentemente, infatti `jES` non ha controllo centralizzato e la ricetta madre è eseguita passando di unit in unit e così si esegue un passo computazionale che fa sì che `OrderDistiller` lanci una ricetta.

Ciò che accade quando la simulazione è lanciata è che la gestione degli ordini nelle unità incontra, durante la lettura delle ricette, il passo computazionale generando un'informazione che sta sospesa in `OrderDistiller` sino al momento di eseguire il prossimo lancio di una ricetta.

Ciò che si è dovuto creare è uno specifico passo che facesse un set in `OrderDistiller` con il codice tratto dalla matrice usata.

La costruzione di questo meccanismo ha posto dei problemi; bisognava infatti capire dove l'`OrderDistiller` andasse a cercare il numero relativo alla ricetta da eseguire. La soluzione più appropriata è sembrata quella di custodire quest'informazione all'interno di un vettore posizionato in `OrderDistiller` stesso. Questo vettore è composto esclusivamente da zeri all'inizio della simulazione, man mano che la simulazione procede, e che i passi computazionali sono eseguiti, esso è riempito con i numeri che corrispondono alle ricette.

`OrderDistiller` legge all'interno del vettore e se trova i numeri delle ricette le esegue e svuota il vettore.

Nello sviluppo del formalismo si è deciso di procedere a piccoli passi in modo da mettere alla prova, volta per volta, la correttezza dei cambiamenti effet-

tuati. Durante la prima fase sono stati creati nuovi passi computazionali inseriti in `ComputationalAssembler.java`³

La prima versione del file svolgeva una funzione piuttosto elementare. Definisce una matrice, ne leggeva l'indirizzo di memoria ed inseriva il numero 287 nelle posizioni di riga e colonna (0,0). Alla fine il contenuto della matrice veniva visualizzato a video per permettere di capire se il passo computazionale fosse stato compiuto o meno.

Tutto questo è stato creato aggiungendo le seguenti righe di codice nel file:

```
public void c1001()
{
if(pendingComputationalSpecificationSet.
  getNumberOfMemoryMatrixesToBeUsed()!=1)
  {
System.out.println("Code -1001 requires three matrix; " +
  pendingComputationalSpecificationSet.
  getNumberOfMemoryMatrixesToBeUsed() +
  " found in order # " +
  pendingComputationalSpecificationSet.
  getOrderNumber());
MyExit.exit(1);
  }
}
```

`mm0=(MemoryMatrix) pendingComputationalSpecificationSet.`

³Si deve copiare `ComputationalAssembler.java` nella cartella principale del programma da `src/`; questo perché il comando `make run` usa le classi contenute in `lib/jesframe.jar`, che sono quelle contenute in `src/`, ma le classi in `/.`, cioè la cartella corrente, sovrascrivono quelle in `jesframe.jar`.

```

    getMemoryMatrixAddress(0);

mm0.setValue(layer,0,0,287.0);
mm0.print();

done=true;
    } // end c1001

```

Il metodo esegue una parte di controllo comune a tutti i passi computazionali⁴ nella quale cambia unicamente il riferimento al passo specifico, viene letto l'indirizzo di memoria della matrice utilizzata (la numero 0); alla matrice è assegnato il valore 287 che è poi scritto a video per verificare il corretto funzionamento del passo.

Inizialmente si era pensato di creare tanti passi computazionali simili a quello appena descritto, con l'unica differenza che invece di visualizzare a video il numero identificativo della ricetta, lo avrebbero scritto nel vettore contenuto

```

4if
(pendingComputationalSpecificationSet.
getNumberOfMemoryMatrixesToBeUsed()!=1)
{
System.out.println("Code -1001 requires three matrix; " +
pendingComputationalSpecificationSet.
getNumberOfMemoryMatrixesToBeUsed() +
' found in order #' +
pendingComputationalSpecificationSet.
getOrderNumber());
MyExit.exit(1);
}

```

in OrderDistiller; in un secondo momento si è invece deciso di modificare leggermente l'impostazione del formalismo.

Si è preferito creare un passo computazionale⁵ che si occupasse unicamente di inserire in più matrici i numeri delle ricette indipendenti che simulano la comunicazione.

L'altro passo interagisce con l'OrderDistiller riempiendo volta per volta il vettore. Quest'ultimo dovrà funzionare per tutte le ricette che permettono di simulare l'interazione tra box ed ambulanze semplicemente cambiando il numero di matrice all'interno del passo stesso.

Per fare ciò però era necessario conoscere la struttura di OrderDistiller, in quanto è questa classe che, attraverso il metodo Distill, si occupa del lancio delle ricette. Di seguito vengono illustrati i metodi contenuti nella classe OrderDistiller.

Dopo il costruttore

```
public OrderDistiller (Zone aZone, int msn, int msl,
                      ListImpl ul,
                      ListImpl eul, ListImpl ol, int tln,
                      ESFrameModelSwarm mo, AssigningTool at)
{ //pt
  super(aZone, msn, msl, ul, eul,
        ol, tln, mo, at);

  unitList=ul;
  endUnitList=eul;
  orderList=ol;
  assigningTool=at; //pt
```

⁵nello specifico si tratta del passo 1001.

```
}
```

è descritto il metodo `distill`.

Tramite questo metodo sono lanciati gli ordini contenuti nelle ricette secondo la sequenza contenuta nell'`OrderStartingSequence` o nell'`OrderSequence`, a seconda delle esigenze delle singole simulazioni.

Il controllo di quale dei due utilizzare avviene subito:

```
if(firstTime == true)
{
currentOrderSequenceWorksheet=orderSequences1;
firstTime = false;
}
else
currentOrderSequenceWorksheet = orderSequences2;
```

La variabile `firstTime` indica se il file da usare è l'`orderStartingSequence` (`orderSequences1`) o l'`orderSequence` (`orderSequences2`). Il metodo effettua altri controlli e solo successivamente arriva alla lettura delle ricette:

```
//Reading the order from the worksheet
readOrderFrom(orderSequenceWorksheet);//see below
//Checking if the recipe code exists
boolean recipeCodeFound = false;
for (int r = 0; r < recipeList.getCount() &&
!recipeCodeFound; r++)
{
aRecipe = (Recipe) recipeList.atOffset(r);
```

```

if(aRecipe.getRecipeCode() == currentOrder[0])
recipeCodeFound = true;
        }

ed al loro lancio:

for(int q = 0; q < currentOrder[1]; q++)
{
orderCount++;
anOrder=newOrder(getZone(),orderCount,
        Globals.env.getCurrentTime(),aRecipe.getLength()
,aRecipe.getRecipeSteps(), eSFrameModelSwarm, endUnitList);

anOrder.setRecipeName(aRecipe.getRecipeName());
// setting the layer (from 0 to totalLayerNumber-1)
if(currentLayer <0 || currentLayer >= totalLayerNumber)
        {
System.out.println("A layer in the sequence is not
        included\n"+"in the
                interval from 0 to totalLayerNumber-1");
MyExit.exit(1);
        }
        anOrder.setOrderLayer(currentLayer);
// add the active orders to the general order list (they will be
// eliminated when dropped in a unit, being finished); this
// list has been introduced for accounting purposes [may be it would
// be better substitute it with a get to the
                units to know their waiting lists]
orderList.addLast(anOrder);

```

```
// sending the order to the first production unit (we
      are acting as the Front //End of the ES)
```

In particolare il metodo deve controllare, ogni volta che trova un ‘;’, se ci sono in coda passi computazionali da eseguire, nel qual caso li manda in esecuzione prima di passare alla riga della sequenza successiva.

Primi risultati:

Il passo computazionale è stato creato inizialmente nel seguente modo: dopo un controllo sul numero delle matrici richieste si è letto l’indirizzo di memoria della matrice utilizzata dal passo (la numero zero); in questa si è inserito nelle posizioni di riga e colonna (0,0) il numero 287, che corrisponde alla ricetta che deve essere lanciata in contemporanea a quella che la richiama. Si è poi fatto scrivere a video il valore presente nella matrice come verifica⁶. All’inizio della simulazione compare immediatamente a video una scritta in cui viene indicato, nell’ordine, il numero di matrice utilizzata, la fase che richiama il passo computazionale e per quante unità temporali lavora la fase in questione. Utilizzando quest’unica ricetta:

```
ex-post1    142408
100 s 1    200 s 23    301 s 1    c1001 1 0 450 s 12
450 s 42    450 s 114          700 s 0    800 s 6
450 s 180;
```

⁶Per visualizzarlo a video si è dovuto impostare true il campo printMatrixes in jesframe.scm

dove era inserito il nostro passo computazionale la simulazione presentava questa scritta:

```
0 405 s 12'
```

che corrisponde al numero della matrice utilizzata dal passo computazionale (0), la fase che richiama il passo (301) ed i tic della fase (1). Questa scritta è data dal gestore degli ordini, che all'inizio del nostro mondo controlla che tutto sia in ordine e rileva anche la presenza di passi computazionali.

In un secondo momento è stata sviluppata la versione definitiva, per cui è stato creato il passo che carica le matrici (c1001) e quelli che caricano il vettore con i numeri letti dalle matrici di memoria precedentemente riempite.

Interazione tra ComputationalAssembler e OrderDistiller:

Il vettore all'interno di OrderDistiller si chiama computationalRecipes ed è creato dal costruttore di OrderDistiller in misura sovrabbondante (1000 elementi), in modo da contemplare la possibilità che ci siano moltissime ricette da lanciare per ogni unità di tempo.

Ovviamente, nel caso in cui non fosse comunque sufficiente a contenere i numeri identificativi di tutte le ricette la simulazione verrà interrotta visualizzando a video il tipo di errore verificatosi.

In particolare, la dichiarazione del vettore avviene nella seguente forma:

```
int[] computationalRecipes;
```

cioè il tipo di variabile che si vuole usare (in questo caso il vettore è composto da numeri interi), le due parentesi quadre che significano che la variabile è un vettore, ed il nome della variabile stessa.

Il vettore dovrà poi essere effettivamente creato e di questo si occupa il costruttore della classe OrderDistiller nella seguente forma:

```
computationalRecipes new Int [1000]
```

cioè si crea la nuova (new) variabile che è di tipo intero (Int) e che è un vettore composto da 1000 elementi. Il vettore verrà subito inizializzato a zero, il che significa che ad ogni elemento del vettore verrà assegnato il valore zero, come si può vedere dal codice sottostante:

```
// CONSTRUCTOR
public OrderDistiller (Zone aZone,
                      int msn, int msl, ListImpl ul,
ListImpl eul, ListImpl ol, int tln,
                      ESFrameModelSwarm mo, AssigningTool at)
{ //pt
super(aZone, msn, msl, ul, eul, ol, tln, mo, at);

computationalRecipes = new int[1000];
for(int i=0; i<1000; i++)
{
    computationalRecipes[i]=0;
}

unitList=ul;
endUnitList=eul;
```

```

orderList=ol;
assigningTool=at; //pt
}

```

Questa operazione viene compiuta perché successivamente il vettore sarà oggetto di controllo ad ogni tic e verranno lanciate tutte le ricette inserite nel vettore; questo finché non viene trovato uno zero, dopodiché il vettore viene riazzerato.

In seguito si è passati alla modifica di ComputationalAssembler in modo da poterlo fare interagire con OrderDistiller.

Le attuali modifiche sono state effettuate per dotare ComputationalAssembler dell'indirizzo di memoria di OrderDistiller, in modo da permettere l'interazione tra le due classi. Il codice corrispondente è:

```

public class ComputationalAssembler extends
        ComputationalAssemblerBasic {

    OrderDistiller
myOrderDistiller;

    /**
     * the constructor for ComputationalAssembler
     */

    public ComputationalAssembler (Zone aZone, OrderDistiller od)
    {
// Call the constructor for the parent class.
super(aZone, od);

```

```
myOrderDistiller= od;  
    }
```

Lo stesso lavoro è stato fatto nella classe `ComputationalAssemblerBasic`. Dopo aver modificato i parametri da passare alla classe `OrderDistiller` si è dovuto modificare anche la classe `ESFrameModelSwarm`; ogni volta che si incontrava le seguenti righe di codice:

```
aComputationalAssembler = new ComputationalAssembler (getZone());
```

si è dovuto aggiungere anche `myOrderDistiller`, in questo modo:

```
aComputationalAssembler = new ComputationalAssembler(getZone()  
    , myOrderDistiller);
```

Dopo queste modifiche abbiamo più problemi di compilazione per quanto riguarda il programma `Swarm`.

A questo punto l'attenzione si è spostata sulla creazione del metodo `setComputationalRecipe`, appartenente alla classe `OrderDistiller`; tale metodo dovrà ogni volta esplorare il vettore sino a trovare la prima posizione messa a 0, nella quale sarà inserito il numero identificativo della ricetta da lanciare.

L'ultimo passo consiste nel controllare se il vettore è pieno al momento del lancio delle ricette, nel qual caso le ricette all'interno del vettore devono essere eseguite.

6.4 La pulizia dei dati

Allo scopo di migliorare la simulazione abbiamo deciso di apportare alcune modifiche ai dati fornitici dalla centrale operativa di Grugliasco.

I dati disponibili per quanto riguarda un numero importante di interventi erano l'ora di inizio della missione di soccorso (l'ora in cui la centrale operativa riceveva una richiesta di aiuto) e l'ora in cui la missione era ultimata, con la registrazione dei dati personali della persona assistita ad opera del personale addetto alla postazione di chiusura.

I dati riguardanti l'ora di inizio delle missioni rispecchiavano appieno la realtà dei fatti. L'ora di chiusura, invece, nella maggioranza dei casi era frutto di un'approssimazione.

Quando la postazione di chiusura si trova a gestire un numero troppo elevato di interventi, il personale si trova costretto a risparmiare tempo per riuscire a far fronte a tutti gli impegni. Per fare ciò la registrazione dei dati dei pazienti e l'ora di chiusura del servizio delle ambulanze non sono registrati puntualmente, ma periodicamente, ogni qualvolta il personale abbia tempo materiale di effettuare la registrazione.

Ne segue che, a seconda dei casi, un'ambulanza si libererà anche parecchi minuti prima di quanto risulta dai registri della centrale operativa e possibilmente sarà già impegnata in una nuova missione operativa nel momento in cui è considerata ufficialmente di nuovo libera.

Per ovviare a questo inconveniente si è deciso di dividere per comodità l'ambito cittadino da quello provinciale, in seguito abbiamo proceduto nel seguente modo:

- abbiamo individuato un numero congruo di interventi per i quali i dati previsti fossero stati archiviati correttamente. Abbiamo cerca-

to questi interventi nella giornata della simulazione e nelle settimane immediatamente successive e precedenti;

- abbiamo individuato su una pianta di Torino le posizioni in cui stazionavano le ambulanze durante gli interventi e quelle dove le stesse si sono trovate ad operare;
- abbiamo calcolato la distanza tra tali posizioni in modo da ottenere le distanze percorse dalle ambulanze.

Le distanze sono state ottenute grazie ad un foglio di calcolo all'interno del quale era stato inserito un programma che calcolava il teorema di Pitagora.

A partire dai tempi degli interventi e dalle distanze percorse il programma era in grado di calcolare la velocità con la quale si muovevano le ambulanze;

- abbiamo diviso in fasce orarie gli interventi analizzati, trovando così una serie di velocità per fascia oraria;
- abbiamo eseguito una media delle velocità per ogni fascia oraria;
- verificato che le velocità medie non si discostavano di molto le una dalle altre, abbiamo optato per eseguire una media giornaliera delle velocità;
- abbiamo moltiplicato la velocità media di intervento per le distanze di intervento relative alla giornata simulata. Abbiamo così ottenuto dei tempi di intervento completi e plausibili.

A seguito di questo procedimento i risultati della simulazione sono migliorati sensibilmente: non si formavano più code.

Intendevamo procedere allo stesso modo per pulire i dati riguardanti gli interventi effettuati nella provincia di Torino, purtroppo non siamo riusciti a

trovare una mappa geografica a dettaglio provinciale.

Il problema è stato ovviato tramite l'utilizzo del sito internet: www.tuttocittà.it. Fornendo a [tuttocittà](http://www.tuttocittà.it) informazioni a proposito del luogo di partenza e del luogo di arrivo delle ambulanze, abbiamo ottenuto le informazioni riguardo alla distanza percorsa.

Una volta ottenute le distanze percorse dalle ambulanze il procedimento applicato ai dati da noi posseduti è stato lo stesso descritto per gli interventi effettuati in Torino.

Un ulteriore problema è sorto a questo punto per quanto riguarda gli interventi effettuati in provincia. I dati raccolti erano spesso di qualità inferiore a quelli cittadini. In provincia le ambulanze stazionano spesso presso le sedi delle associazioni di volontariato, conoscere l'indirizzo di queste sedi non è sempre agevole.

Fortunatamente è stato quasi sempre possibile desumere le informazioni cercate.

Ciò non è stato possibile quando ci si trovava di fronte ad interventi che partivano da cittadine talmente piccole da non fare una grossa differenza se le ambulanze fossero partite dalla sede dell'associazione o da qualsiasi altro punto all'interno del paese.

In questi casi si considerava che la missione partisse dall'indirizzo corrispondente al municipio.

Un discorso analogo va per i luoghi dove le ambulanze dovevano recarsi a prestare soccorso: questi erano spesso indicati genericamente.

Fortunatamente si è quasi sempre riusciti a risalire ad un indirizzo utile e gli interventi scartati sono stati veramente pochi.

Alla fine la velocità media provinciale è risultata superiore a quella cittadina, ma non in maniera elevata, con una distribuzione molto differente tra i piccoli

centri rurali e le cittadine facenti parte della cintura di Torino (tanto da farci ritenere che sarebbe stato meglio includere queste nel modello cittadino).

6.5 Un primo modello ex-ante

Con la messa a punto del modello ex-post e la pulizia dei dati lo scopo del lavoro è cambiato. La simulazione riproduceva ormai fedelmente la realtà, si trattava ora di utilizzare il modello per ipotizzare dei cambiamenti all'interno della realtà organizzativa del 118, provarli sul computer e vagliare il risultato ottenuto.

I possibili cambiamenti all'interno di un'organizzazione sono tantissimi, provarli a caso può risultare poco costruttivo. Il procedimento più logico è scoprire quali sono i punti più sensibili dell'organizzazione (i punti in corrispondenza dei quali la produzione del servizio rallenta eccessivamente a fronte di un aumento importante delle richieste di assistenza) e operarvi dei cambiamenti.

Per ottenere questo risultato è necessario sovraccaricare il modello (far sì che debba processare carichi di lavoro molto più grandi di quelli che normalmente si sono presenti in una giornata normale) e osservare in corrispondenza di quali unità produttive si formano le code di attesa più grandi.

Per poterlo fare è però necessario passare prima dal modello ex-post ad un modello ex-ante.

Nel modello ex-post [vedi supra] ogni unità produttiva si limitava a svolgere un compito specifico che le era effettivamente appartenuto durante la giornata simulata.

Per esempio, se l'ambulanza 835 aveva effettuato una missione di soccorso alle ore 14:35, anche nella simulazione avrebbe dovuto ripetere lo stesso intervento. Se fossero state libere al momento due ambulanze abilitate ad effettuare lo stesso tipo di interventi, il lavoro sarebbe dovuto comunque essere svolto dall'ambulanza 835.

Nel modello ex-ante, invece, le unità produttive non devono limitarsi a svolg-

Simple production unit data are reported in a text file
(unitData/unitBasicData.txt)

Figure 8

unit_#	useWarehouse	prod.phase_#	fixed_costs	variable_costs
1	1	11	12	1
2	1	0	0	0
3	1	3	15	2
4	1	0	0	0
5	1	51	12	2
6	1	6	11	20
7	1	12	23	1
8	1	8	22	11
9	1	13	7	12
10	1	18	40	7
11	1	11	5	1

Figura 6.5: Il file unitBasicData.txt.

re un compito solo (come saper intervenire una volta alle ore 14:35 e un'altra alle ore 17:58), ma sono in grado di svolgere diversi compiti.

Per introdurre queste unità 'complesse' bisogna effettuare una serie di modifiche all'interno del jesframe. Prima di tutto è necessario modificare il file unitBasicData.txt.

Nella colonna intitolata production phase, invece di inserire il numero relativo al compito specifico che l'unità è in grado di eseguire, è necessario inserire il numero '0'.

L'esistenza di numeri '0' all'interno di unitBasicData.txt obbliga il programma ad andare alla ricerca del file units.xls (custodito come unit BasicData.txt all'interno della cartella unitData).

La creazione del file units.xls è la seconda modifica necessaria per creare un modello ex-ante. Questo file contiene un foglio di calcolo per ogni unità

A specific complex unit

Figure 10

	A	B	C	D	E	F	G	H	I	J	K
1	3										
2											
3		201	1	1	0						
4		2001	1	1	0						
5		2	1	1	1						
6											
7											
8											
9											
10											
11											
12											
13											
14											
15											
16											
17											
18											
19											
20											
21											
22											
23											
24											
25											
26											

Figura 6.6: Il file units.xls.

complessa presente nel modello. Ogni foglio di calcolo ha il nome dell'unità complessa che rappresenta. Il primo dato che figura è un numero relativo a quante operazioni l'unità è capace di svolgere.

Nella figura in alto l'unità sa svolgere tre compiti: l'operazione 201, l'operazione 2001 e l'operazione 2.

Gli altri numeri presenti nel foglio di calcolo si riferiscono ai compiti che l'unità è capace di portare a termine (201, 2001 e 2), ai costi fissi che le unità devono affrontare per la produzione (1,1,1), ai costi variabili che dette unità devono sostenere per poter produrre (1,1,1) e alla possibilità che l'unità stia producendo un inventario o meno (0 se l'unità non sta producendo un inventario, 1 in caso contrario). Queste informazioni sono riassunte nella figura relativa allo schema generale di un'unità complessa.

Per passare al modello ex-ante, nel file units.xls relativo alla simulazione del

The general scheme, to be used as a remark page

Figure 9

	A	B	C	D	E	F	G	H
1	nOfPhasesToDealWith							
2								
3	phase 1	fixed costs 1	variable costs 1	inventories in production 1				
4	phase 2	fixed costs 2	variable costs 2	inventories in production 2				
5	...							
6	phase n	fixed costs n	variable costs n	inventories in production n				
7								
8	sc 1 1	sc 1 2	...	sc 1 n				
9	sc 2 1	sc 2 2	...	sc 2 n				
10				
11	sc n 1	sc n 2	...	sc n n				
12								
13	st 1 1	st 1 2	...	st 1 n				
14	st 2 1	st 2 2	...	st 2 n				
15				
16	st n 1	st n 2	...	st n n				
17								
18	Remarks							
19	hopefully, fixed costs are the same for all phases,							
20	anyway, when the unit state is undefined (no production made) we use the fixed costs of the first row							
21								
22	inventory production can be 0 (no) or 1 (yes); normally, only one row is set to 1							
23	if we find more than one row set to 1, the first one is chosen							
24								
25	sc ij = setup costs from state i to state j				st ij = setup time from state i to state j			
26								

Figura 6.7: Lo schema generale di un'unità complessa.

118 sono state introdotte una serie di unità complesse rappresentanti le ambulanze che intervengono per prestare soccorso.

Ad ogni ambulanza è stata data la possibilità di svolgere due tipi di compiti diversi: i compiti che aveva effettivamente svolto durante la giornata simulata e quelli attinenti alla sua tipologia di ambulanza (per esempio le ambulanze meglio equipaggiate, le MSA, potevano intervenire per ripetere gli interventi da loro effettuati durante la giornata oggetto di simulazione oppure potevano svolgere le mansioni che erano state proprie di un'altra ambulanza MSA)

Introdotte queste modifiche nel modello si è proceduto come segue:

- per prima cosa abbiamo fatto funzionare il modello chiedendo ad ogni ambulanza di ripetere gli interventi effettivamente svolti durante la giornata oggetto di simulazione (ad ulteriore conferma della solidità e della bontà del modello). Coerentemente con le nostre previsioni non

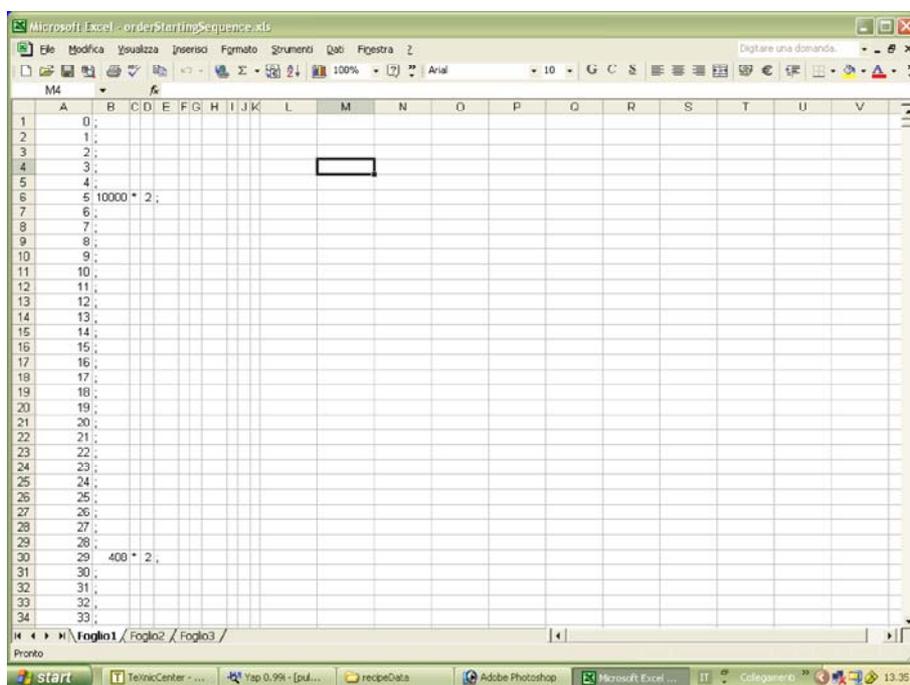


Figura 6.9: Il file orderStartingSequence.xls.

Questo risultato è stato ottenuto agendo sul file orderStartingSequence.xls.

Il file orderStartingSequence.xls contiene le informazioni riguardanti quali interventi avvengono in una giornata e a che ora essi si verificano.

Per sovraccaricare il modello è stato sufficiente raddoppiare gli interventi previsti per la simulazione iniziale.

I risultati hanno dimostrato che le prime unità produttive ad andare in crisi sono le PVS e la postazione di chiusura.

Abbiamo allora pensato di introdurre una serie di unità PVS in più, per verificare in che misura un aumento della capacità produttiva dell'organizzazione, concentrato nel punto più esposto ad un rallentamento nell'erogazione del servizio, potesse far fronte alle code verificatesi.

Nella realtà esistono sei postazioni di valutazione sanitaria.

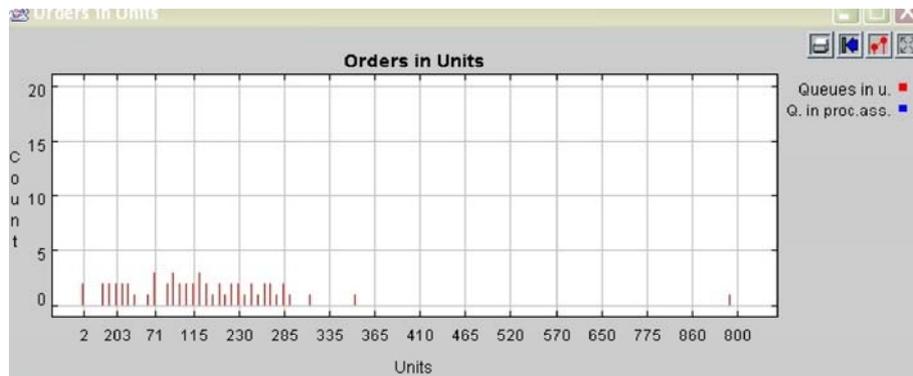


Figura 6.10: Il modello in condizioni normali.

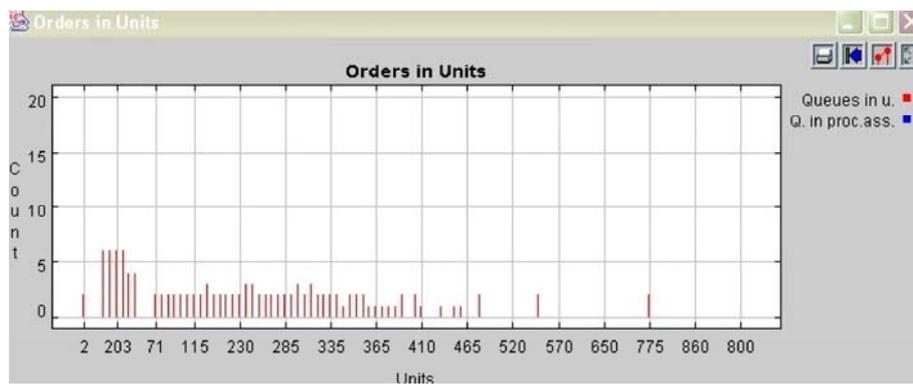


Figura 6.11: Il modello sovraccaricato.

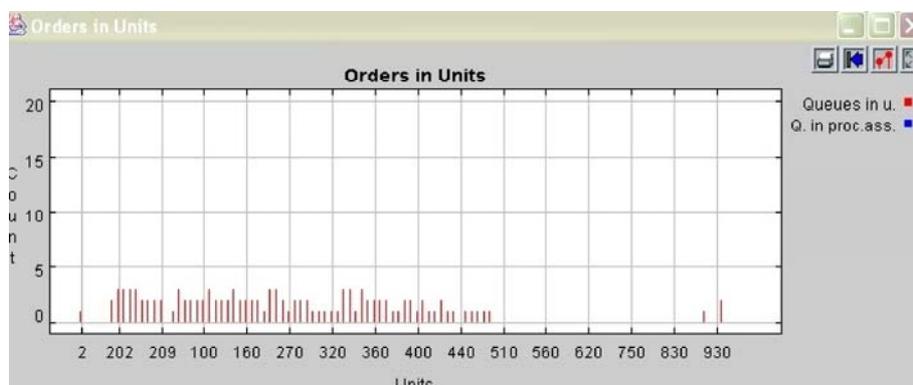


Figura 6.12: Le code si normalizzano con l'introduzione di nuove PVS.

Teoricamente, a fronte di un aumento degli interventi pari al totale degli interventi effettuati in una giornata, bisognerebbe raddoppiare le PVS per essere sicuri che il servizio sia erogato in maniera ottimale.

La simulazione ha invece mostrato che basta introdurre tre nuove PVS perché la situazione si normalizzi e le code scompaiano.

È perciò possibile far fronte ad una situazione di eccezionale emergenza senza essere costretti ad assumere un numero elevato di nuovi operatori.

Questo fatto è interessante in quanto mette in luce uno dei possibili vantaggi del metodo simulativo. Il servizio di pronto soccorso si occupa di assistere persone che sono rimaste vittime di incidenti o che hanno accusato dei malori. Il pronto intervento di infermieri, medici e volontari riesce in molti casi a scongiurare che le conseguenze degli incidenti e dei malori siano fatali.

Un servizio così importante, dove un ritardo o un disguido possono avere conseguenze pesantissime, non può permettersi di effettuare troppi esperimenti per migliorare il servizio. Ogni cambiamento porta con sé incertezze e contrattempi.

La simulazione può servire da guida nel processo di miglioramento del servizio fornito ai cittadini. I cambiamenti non saranno così un'incognita. Se

un cambiamento nella simulazione si è rivelato inutile o dannoso ai fini della qualità finale del servizio sarà scartato a priori. Se, invece, un cambiamento si sarà dimostrato responsabile di un miglioramento nel funzionamento dell'organizzazione, si potrà valutare, con cautela, l'ipotesi di introdurlo nella realtà.

Il caso concreto è importante per il seguente motivo: per essere sicuri di offrire la stessa qualità del servizio in caso di forte emergenza, la centrale operativa, se non fosse a conoscenza dei risultati della simulazione, sarebbe costretta ad assumere sei nuovi operatori.

La simulazione dimostra che per ottenere lo stesso risultato di nuovi operatori ne bastano tre, con un evidente risparmio sia in termini economici, ma soprattutto di coordinamento tra i nuovi operatori ed il personale già presente.

È utile aggiungere che in un'organizzazione come il 118, dove la qualità del servizio è estremamente importante, è utile che i cambiamenti siano i più mirati e della minore entità possibili, per non stravolgere il normale funzionamento della centrale operativa con evidente rischio che si verifichino dei disguidi.

6.6 Impegni per il futuro.

Alcuni aspetti del modello devono essere ancora sviluppati. In futuro diventerà importante introdurre alcune modifiche che permettano di rendere la simulazione ancora più realistica e capace di effettuare previsioni che aiutino l'organizzazione del 118 a migliorare il servizio offerto.

Gli aspetti che saranno introdotti in futuro nel modello sono i seguenti:

- diversa disposizione delle ambulanze sul territorio. In futuro sarà possibile visualizzare la disposizione spaziale delle ambulanze all'interno delle diverse aree di intervento;
- diverse squadre con mezzi in grado di muoversi più agilmente nel traffico. Secondo l'idea del dottor Ghiselli, direttore della centrale operativa di Grugliasco, potrebbe essere utile introdurre delle autovetture di soccorso a fianco delle ambulanze. Le autovetture servirebbero per portare sul luogo di un incidente medici e personale specializzato.
Le ambulanze di base, raggiunte sul luogo dell'incidente dalle autovetture di soccorso, permetterebbero di ricomporre in loco equipaggi di cui finora dispongono solo le ambulanze avanzate.
Le autovetture sono più agili e veloci delle ambulanze, ragione per cui si ricaverebbe un vantaggio in termini di copertura del territorio da parte di medici e personale specializzato;
- possibilità di ridefinire la missione di un'ambulanza già destinata. In caso di emergenza un'ambulanza già impegnata in una missione di soccorso può essere reindirizzata verso un altro intervento. Questo caso si verifica soprattutto se il primo intervento è effettuato per prestare soccorso in caso di malori di lievi entità;

- scelta dell'ospedale di destinazione. Spesso l'ospedale di destinazione è una scelta obbligata. In caso di serio trauma l'infortunato sarà portato al C.T.O., in caso di infarto sarà necessario recarsi presso un ospedale dotato di unità coronarica;
- controlli per evitare di mandare più mezzi nello stesso luogo per evitare di sprecare inutilmente risorse.

Capitolo 7

Conclusioni

Un'organizzazione costituisce un esempio di sistema complesso. Non è facile prevedere l'andamento di un sistema così composto a partire dal suo comportamento nel passato o capire in che misura i singoli elementi di cui esso si compone influiscano sul risultato finale.

Nell'organizzazione i diversi operatori, impiegati e dirigenti interagiscono tra loro per formare un quadro di insieme tutt'altro che semplice.

Una possibilità per studiare le relazioni complesse è data dalla metodologia ad agenti. In una simulazione di questo tipo gli agenti (sotto forma di codice informatico) interagendo tra di loro in un ambiente simulato (la realtà dell'organizzazione) concorrono a determinare il comportamento generale del sistema.

Nel caso del 118, gli agenti rappresentano gli operatori che lavorano presso la centrale operativa di Grugliasco e i medici, gli infermieri e i volontari che prestano servizio di soccorso come equipaggio delle differenti ambulanze.

I vantaggi che la simulazione ad agenti può portare al servizio del 118 sono importanti. In una qualsiasi organizzazione è fondamentale offrire il miglior servizio possibile. Per farlo può essere necessario introdurre dei cambiamenti

nella struttura organizzativa. I cambiamenti devono però essere attentamente vagliati.

La simulazione ad agenti permette di ricreare dentro al computer la realtà dell'organizzazione comprensiva dei cambiamenti ipotizzati. Le conseguenze dei cambiamenti emergeranno come risultato del funzionamento della simulazione.

Grazie al programma jES è stato possibile effettuare uno studio del '118'. Nella simulazione è stato ipotizzata la presenza di un'emergenza che facesse aumentare in modo importante gli interventi che il servizio di pronto soccorso effettua ogni giorno.

Sono state così individuate le parti più sensibili dell'organizzazione (quelle che andavano più facilmente in affanno a causa dell'aumento di interventi) ed è stato determinato in che misura esse sarebbero dovute essere rafforzate. Il risultato della simulazione si è rivelato interessante poiché ha mostrato che la struttura operativa del 118 possiede una flessibilità tale da permetterle di affrontare carichi di lavoro molto più impegnativi di quelli medi.

L'unico intervento che si è reso necessario è stato volto ad aumentare la 'capacità produttiva' delle postazioni di valutazione sanitaria.

Raddoppiando i carichi di lavoro giornalieri medi, un semplice aumento di tre operatori a fronte dei sei che operano normalmente è stato sufficiente per controllare la situazione.

Appendice A

Il Decreto del 27 marzo 1992.

Decreto del Presidente della Repubblica 27 marzo 1992

Atto di indirizzo e coordinamento alle regioni per la determinazione dei livelli di assistenza sanitaria di emergenza

(G.U. 31 marzo 1992, n.76, serie generale)

IL PRESIDENTE DELLA REPUBBLICA

Visto l'art. 4 della legge 30 dicembre 1991, n. 412, che detta norme in materia di assistenza sanitaria per l'anno 1992;

Visto il comma 1 della richiamata norma che autorizza il Governo ad emanare un atto di indirizzo e di coordinamento per la determinazione dei livelli di assistenza sanitaria da assicurare in condizioni di uniformità sul territorio nazionale sulla base dei limiti e principi di cui alle successive lettere a), b), c), d) ed e);

Vista la deliberazione del CIPE in data 3 agosto 1990 che ha disciplinato, su conforme parere della Conferenza permanente per i rapporti tra lo Stato, le regioni e le province autonome, le priorità degli interventi relativi all'emergenza-urgenza sanitaria ed al rischio anestesiologicalo anche utilizzando con vincolo di destinazione le risorse in conto capitale del Fondo sanitario nazionale;

Visto l'art. 22 dell'accordo collettivo nazionale per la regolamentazione dei rapporti con i medici addetti al servizio di guardia medica e di emergenza territoriale, reso esecutivo con decreto del Presidente della Repubblica 25 gennaio 1991, n. 41;

Visto il documento tecnico di intesa approvato dalla Conferenza Stato-regioni nella seduta del 14 gennaio 1992;

Visto il parere espresso dal Consiglio superiore di sanità in data 12 febbraio 1992;

Ritenuto che, nelle more della definizione degli standard organizzativi e dei costi unitari dei livelli di assistenza uniformi di cui all'art. 4 della legge 30 dicembre 1991, n. 412, la Conferenza Stato-regioni in data 7 febbraio 1992 ha definito l'intesa sul livello uniforme di assistenza del sistema dell'emergenza sanitaria;

Ritenuto che le spese in conto capitale per l'organizzazione del livello assistenziale fanno carico agli stanziamenti di cui all'art. 20 della legge 11 marzo 1988, n. 67, nonché agli stanziamenti in conto capitale del Fondo sanitario nazionale, mentre quelle correnti fanno carico al Fondo sanitario nazionale di parte corrente di cui all'art. 51 della legge 23 dicembre 1978, n. 833, nella misura che sarà determinata ai sensi del combinato disposto della norma di cui ai commi 1 e 16 dell'art. 4 della legge 30 dicembre 1991, n. 412;

Vista la deliberazione del Consiglio dei Ministri, adottata nella riunione del 13 marzo 1992, su proposta del Ministro della sanità, di concerto con il Ministro per le riforme istituzionali e gli affari regionali;

DECRETA:

È approvato il seguente atto di indirizzo e coordinamento delle attività

delle regioni e delle province autonome di Trento e di Bolzano, in materia di emergenza sanitaria.

Art. 1 Il livello assistenziale di emergenza sanitaria

1. Ai sensi del comma 1 dell'art. 4 della legge 30 dicembre 1991, n. 412, il livello assistenziale di emergenza sanitaria da assicurare con carattere di uniformità in tutto il territorio nazionale è costituito dal complesso dei servizi e delle prestazioni di cui agli articoli successivi.

Art. 2 Il sistema di emergenza sanitaria

1. Le regioni e le province autonome di Trento e di Bolzano organizzano le attività di urgenza e di emergenza sanitaria articolate su:

- a. il sistema di allarme sanitario;
- b. il sistema di accettazione e di emergenza sanitaria.

Art. 3 Il sistema di allarme sanitario

1. Il sistema di allarme sanitario è assicurato dalla centrale operativa, cui fa riferimento il numero unico telefonico nazionale '118'. Alla centrale

operativa affluiscono tutte le richieste di intervento per emergenza sanitaria. La centrale operativa garantisce il coordinamento di tutti gli interventi nell'ambito territoriale di riferimento.

2. Le centrali operative della rete regionale devono essere compatibili tra loro e con quelle delle altre regioni e delle province autonome di Trento e di Bolzano in termini di standard telefonici di comunicazione e di servizi per consentire la gestione del traffico interregionale. Con decreto del Ministro della sanità, di concerto con il Ministro delle poste e delle telecomunicazioni, entro sessanta giorni dalla data di pubblicazione del presente atto nella Gazzetta Ufficiale della Repubblica, sono definiti gli standard di comunicazione e di servizio.

3. L'attivazione della centrale operativa comporta il superamento degli altri numeri di emergenza sanitaria di enti, associazioni e servizi delle unità sanitarie locali nell'ambito territoriale di riferimento, anche mediante convogliamento automatico delle chiamate sulla centrale operativa del '118'.

4. Le centrali operative sono organizzate, di norma su base provinciale. In ogni caso nelle aree metropolitane dove possono all'occorrenza sussistere più centrali operative, è necessario assicurare il coordinamento tra di esse.

5. Le centrali operative assicurano i radiocollegamenti con le autoam-

bulanze e gli altri mezzi di soccorso coordinati e con i servizi sanitari del sistema di emergenza sanitaria del territorio di riferimento, su frequenze dedicate e riservate al servizio sanitario nazionale, definite con il decreto di cui al comma 2.

6. Il dimensionamento e i contenuti tecnologici delle centrali operative sono definiti sulla base del documento approvato dalla Conferenza Stato-regioni in data 14 gennaio 1992, che viene allegato al presente atto.

Art. 4 Competenze e responsabilità nelle centrali operative

1. La responsabilità medico-organizzativa della centrale operativa è attribuita nominativamente, anche a rotazione, a un medico ospedaliero con qualifica non inferiore ad aiuto corresponsabile, preferibilmente anestesista, in possesso di documentata esperienza ed operante nella medesima area dell'emergenza.

2. La centrale operativa è attiva per 24 ore al giorno e si avvale di personale infermieristico adeguatamente addestrato, nonché di competenze mediche di appoggio. Queste devono essere immediatamente consultabili e sono assicurate nominativamente anche a rotazione, da medici dipendenti con esperienza nel settore dell'urgenza ed emergenza e da medici del servizio di guardia medica di cui all'art. 22 dell'accordo collettivo nazionale per la rego-

lamentazione dei rapporti con i medici addetti al servizio di guardia medica e di emergenza territoriale, reso esecutivo con decreto del Presidente della Repubblica 25 gennaio 1991, n. 41. La responsabilità operativa è affidata al personale infermieristico professionale della centrale, nell'ambito dei protocolli decisi dal medico responsabile della centrale operativa.

Art. 5 Disciplina delle attività

1. Gli interventi di emergenza sono classificati con appositi codici. Il Ministro della sanità, con proprio decreto da emanarsi entro sessanta giorni dalla data di pubblicazione del presente atto nella Gazzetta Ufficiale/e della Repubblica, stabilisce criteri e requisiti cui debbono attenersi le regioni e le province autonome di Trento e di Bolzano nella definizione di tale codificazione, anche ai fini delle registrazioni necessarie per documentare le attività svolte e i soggetti interessati.

2. L'attività di soccorso sanitario costituisce competenza esclusiva del Servizio sanitario nazionale. Il Governo determina gli standard tipologici e di dotazione dei mezzi di soccorso ed i requisiti professionali del personale di bordo, di intesa con la Conferenza Stato-regioni.

3. Ai fini dell'attività di cui al precedente comma, le regioni e le province autonome di Trento e di Bolzano possono avvalersi del concorso di enti e

di associazioni pubbliche e private, in possesso dell'apposita autorizzazione sanitaria, sulla base di uno schema di convenzione definito dalla Conferenza Stato-regioni, su proposta del Ministro della sanità.

Art. 6 Il sistema di accettazione e di emergenza sanitaria

1. Fermo restando quanto previsto dall'art. 14 del decreto del Presidente della Repubblica 27 marzo 1969, n. 128, in materia di accettazione sanitaria, il sistema di emergenza sanitaria assicura:

- a. il servizio di pronto soccorso;
- b. il dipartimento di emergenza.

2. Le regioni e le province autonome di Trento e di Bolzano individuano gli ospedali sedi di pronto soccorso e di dipartimento di emergenza.

Art. 7 Le funzioni di pronto soccorso

1. L'ospedale sede di pronto soccorso deve assicurare, oltre agli interventi diagnostico-terapeutici di urgenza compatibili con le specialità di cui è dotato, almeno il primo accertamento diagnostico, clinico, strumentale e di laboratorio e gli interventi necessari alla stabilizzazione del paziente, nonché garantire il trasporto protetto.

2. La responsabilità delle attività del pronto soccorso e il collegamento con le specialità di cui è dotato l'ospedale sono attribuiti nominativamente, anche a rotazione non inferiore a sei mesi, ad un medico con qualifica non inferiore ad aiuto, con documentata esperienza nel settore.

Art. 8 Le funzioni del dipartimento di emergenza

1. il dipartimento di emergenza deve assicurare nell'arco delle 24 ore, anche attraverso le unità operative specialistiche di cui è dotato l'ospedale, oltre alle funzioni di pronto soccorso, anche:

- a. interventi diagnostico-terapeutici di emergenza medici, chirurgici, ortopedici, ostetrici e pediatrici;
- b. osservazione breve, assistenza cardiologica e rianimatoria.

2. Al dipartimento di emergenza sono assicurate le prestazioni analitiche, strumentali e di immunoematologia per l'arco delle 24 ore giornaliere .

3. La responsabilità delle attività del dipartimento e il coordinamento con le unità operative specialistiche di cui è dotato l'ospedale sono attribuiti nominativamente, anche a rotazione non inferiore a sei mesi, ad un primario medico, chirurgo o rianimatore, con documentata esperienza nel settore.

Art. 9 Le funzioni regionali

1. Le regioni e le province autonome di Trento e di Bolzano, anche a stralcio del Piano sanitario regionale, determinano, entro centoventi giorni, dalla data di pubblicazione del presente atto nella Gazzetta Ufficiale della Repubblica, la ristrutturazione del sistema di emergenza sanitaria, con riferimento alle indicazioni del parere tecnico fornito dal Consiglio superiore di sanità, in data 12 febbraio 1991, e determinano le attribuzioni dei responsabili dei servizi che compongono il sistema stesso.

Il provvedimento di cui al comma precedente determina altresì le modalità di accettazione dei ricoveri di elezione in relazione alla esigenza di garantire adeguate disponibilità di posti letto per l'emergenza. Con il medesimo provvedimento sono determinate le dotazioni di posti letto per l'assistenza subintensiva da attribuire alle singole unità operative.

Art. 10 Prestazioni del personale infermieristico

1. Il personale infermieristico professionale, nello svolgimento del servizio di emergenza, può essere autorizzato a praticare iniezioni per via endovenosa e fleboclisi, nonché a svolgere le altre attività e manovre atte a salvaguardare le funzioni vitali, previste dai protocolli decisi dal medico responsabile del servizio.

Art. 11 Onere del trasporto di emergenza

1. Gli oneri delle prestazioni di trasporto e soccorso sono a carico del servizio sanitario nazionale solo se il trasporto è disposto dalla centrale operativa e comporta il ricovero del paziente. Detti oneri sono altresì a carico del Servizio sanitario nazionale anche in mancanza di ricovero determinata da accertamenti effettuati al pronto soccorso.

Art. 12 Attuazione

1. All'attuazione di quanto disposto dal presente atto provvedono le regioni e le province autonome.

2. Le spese in conto capitale per l'organizzazione del livello assistenziale fanno carico come priorità agli stanziamenti di cui all'art. 20 della legge 11 marzo 1988, n. 67, nonché agli stanziamenti in conto capitale del Fondo sanitario nazionale, mentre quelle correnti fanno carico al Fondo sanitario nazionale di parte corrente di cui all'art. 51 della legge 23 dicembre 1978, n. 833, nella misura che sarà determinata al sensi del combinato disposto delle norme di cui ai commi 1 e 16 dell'art. 4 della legge 30 dicembre 1991, n. 412.

3. Entro sei mesi dalla data di pubblicazione del presente atto nella Gazzetta Ufficiale della Repubblica, la Conferenza Stato regioni verifica le iniziative assunte, lo stato di attuazione del sistema emergenza sanitaria in ciascuna regione e provincia autonoma, nonché le risorse finanziarie impiegate. Allo scopo di attuare il sistema di emergenza sanitaria nelle regioni che non lo abbiano attuato, in tutto o in parte la Conferenza Stato-regioni approva uno schema tipo di accordo di programma, che, sottoscritto dal Ministro della sanità e dal Presidente della regione interessata, determina tempi, modi e risorse finanziarie per l'attuazione, anche avvalendosi di apposite conferenze dei servizi. L'accordo di programma può essere attivato anche prima della verifica, su richiesta della regione e provincia autonoma.

Il presente decreto sarà pubblicato
nella Gazzetta Ufficiale della Repubblica
italiana.

Bibliografia

- [1] Robert Axtell. Why agents? on the varied motivations for agent computing in the social sciences. *Center on Social and Economic Dynamics Working Paper No.17*, November 2000.
- [2] Carlo Bernardini. Caos nel caos. *La rivista dei libri*, Maggio 1994.
- [3] J.L. Borges. *Borges: Cuentos*. Kapelusz, 1995.
- [4] M. Capek. *The Philosophical impact of Contemporary Physics*. Princeton, 1961.
- [5] F. Capra. *Il Tao della fisica*. gli Adelphi, 1982.
- [6] Stock e Castelfranchi. *Macchine come noi. La scommessa dell'intelligenza artificiale*. Laterza, 2000.
- [7] M. e P.S. Churchland. Può una macchina pensare? *Le scienze*, Marzo 1990.
- [8] Epstein and Axtell. Agent-based modeling: Understanding our creations. *The Bulletin of The Santa Fe Institute*, 1994.
- [9] Gianluigi Ferraris. Algoritmi genetici e classifier system nei modelli di agenti. <http://www.swarm.org/comunity-contrib.html>, Marzo 2000.

- [10] Robert Gibbons. Why organizations are such a mess (and what an economist might do about it). <http://web.mit.edu/rgibbons/www/>, March 2000.
- [11] Gilbert and Terna. How to build and use agent-based models in social science. *Mind & Society*, 1, 2000.
- [12] D.R. Hofstadter. *Gödel, Escher, Bach: un'Eterna Ghirlanda Brillante*. gli Adelphi, 1984.
- [13] J. Holland. *Adaptation in neural artificial systems*. Ann Arbor, 1975.
- [14] J. Holland. *Hidden Order: How adaptation builds complexity*. Addison-Wesley, 1995.
- [15] H.E. Kilpatrick. Complexity, spontaneous order and friederich hayek: are spontaneous order and complexity essentially the same thing? *COMPLEXITY*, 2001.
- [16] Alan Kirman. Ants, rationality and recruitment. *The Quarterly Journal of Economics*, Vol. 108, No. 1, February 1993.
- [17] Sergio Margarita. Verso un «robot oeconomicus»: Algoritmi genetici ed economia. *Sistemi Intelligenti*, Dicembre 1992.
- [18] P. Omerod. *L'economia della farfalla. Società mercato e comportamento*. Instar Libri, 2003.
- [19] Domenico Parisi. *Simulazioni. La realtà rifatta nel computer*. Il Mulino, 2001.
- [20] J.Barkley Rosser. On the complexities of complex economic dynamics. *Journal of Economic Perspectives* vol.13, no. 4, 1999.

- [21] David Ruelle. *Hasard et Chaos*. Éditions Odile Jacob, 1991.
- [22] Roberto Satolli. Un cadavere nella stanza cinese. *La rivista dei libri*, Dicembre 1999.
- [23] J.R. Searle. La mente è un programma? *Le scienze*, Marzo 1990.
- [24] H. Simon. *Administrative Behavior*. The Free Press, fourth edition, 1997.
- [25] H. Simon. *Scienza economica e comportamento umano*. Edizioni di Comunità, 2000.
- [26] Ian Stewart. L'angolo matematico. *Le Scienze n. 313*, Settembre 1994.
- [27] Pietro Terna. Reti neurali artificiali e modelli con agenti adattivi. *XXXV Riunione Scientifica Annuale della Società Italiana degli Economisti*, Ottobre 1994.
- [28] Pietro Terna. Simulation tools for social scientists: Building agent based models with swarm. *Journal of Artificial Societies and Social Simulation vol. 1, no. 2*, 1998.
- [29] Pietro Terna. Consumatori e mercato. *Atlante del XXI secolo*, 2001.
- [30] Pietro Terna. The quest for the enterprise: jes, a java enterprise simulator. *disponibile presso <http://web.econ.unito.it/terna>*, August 2003a.
- [31] Pietro Terna. How to use java enterprise simulator (jes) program. *disponibile presso <http://web.econ.unito.it/terna>*, September 2003b.
- [32] M.M. Waldrop. *Complessità. Uomini e idee al confine tra ordine e caos*. Instar Libri, 1995.