Università degli studi di Torino

Facoltà di Economia
Corso di laurea in Economia e Commercio

Tesi di laurea

Simulazione ad agenti e realtà nello studio di bolle e *crash* dei mercati di borsa.

Relatore:
Chiar.mo Prof. Pietro Terna

Correlatore:
Chiar.mo Prof. Sergio Margarita

Candidato:
Andrea Bosio

Anno Accademico 2001-2002

I

Introduzione

Secondo la teoria economica gli operatori finanziari sono razionali ed adottano schemi di comportamento che ottimizzano le loro funzioni di utilità. Nella loro azione accedono a tutta l'informazione disponibile e compiono le loro scelte attraverso la formazione di aspettative "razionali". Ciò dovrebbe avvenire in tutti i mercati e in special modo in quelli finanziari.

La ormai celebre "esuberanza irrazionale" che ha caratterizzato la dinamica delle quotazioni dei titoli azionari nel biennio 1999-2000, al culmine di un lungo periodo di espansione, potrebbe rientrare tra quelle anomalie che la teoria classifica come eccezioni. Eppure, sono frequenti, sui mercati finaziari, periodi in cui i prezzi sembrano divergere sensibilmente dal valore intrinseco dei titoli. Queste fasi di mercato sono definite dagli studiosi *bolle speculative* e sono solitamente seguite da più o meno rapidi ritorni dei corsi verso le valutazioni corrette delle attività (*crash*).

Al culmine della bolla, l'euforia è massima e gli operatori hanno fiducia che i prezzi continueranno a salire e che nuovi capitali continueranno ad entrare sul mercato, alimentando la crescita. Le loro azioni incorporano queste attese, innescando una serie di retroazioni tra i fattori che sostengono lo sviluppo della bolla. Componenti psicologiche e sociali spingono gli investitori a domandare gruppi ritretti di titoli, anche a causa di comportamenti di imitazione.

Solo la successiva percezione di insostenibilità dei prezzi raggiunti, sensibilmente lontani dal loro valore, spinge i primi operatori a iniziare a vendere le attività detenute, realizzando plusvalenze. Ancora una volta, l'imitazione favorisce il diffondersi di questi comportamenti e di queste valutazioni, inducendo sempre più operatori a modificare le loro decisioni di investimento: il crescere della pressione in vendita sui titoli innesca veloci ribassi, alimentando nuove vendite. Il panico inizia a dilagare e la caduta dei corsi si fa sempre più veloce.

Il lavoro che viene presentato tenta una analisi di questi fenomeni, partendo dalla loro spiegazione nella teoria dei mercati finanziari classici per giungere ad una loro ricostruzione sperimentale.

Il primo capitolo descrive le ipotesi sottostanti la teoria di efficienza dei mercati, nelle sue varie accezioni più o meno forti. In questa sezione sono presentati alcuni modelli di valutazione dei prezzi in funzione di diversi gradi di disponibilità delle informazioni. Questa introduzione consente l'analisi e la verifica di alcuni modelli teorici proposti per spiegare la formazione delle bolle speculative. La loro capacità di rappresentare i fenomeni reali, tuttavia, si è spesso mostrata solo parziale . Particolare attenzione è dedicata al modello descrittivo proposto da Shiller, capace di individuare i molteplici fattori acceleranti e meccanismi amplificatori dalla cui azione e retroazione si sarebbe sviluppata la bolla speculativa del biennio 1999-2000.

Se il compito dello studioso è quello di comprendere e conoscere la realtà, non solo attraverso l'osservazione, ma anche con la ricreazione artificiale in laboratorio, l'impossibilità di studiare bolle e *crash* in maniera sperimentale ha costituito un limite ai progressi nella teoria dei mercati finanziari. Lo sviluppo e la diffusione dell'elaboratore elettronico ha reso possibile la nascita di un metodo innovativo di studio e di analisi della realtà, la simulazione, strumento presentato e discusso nel capitolo secondo. Questa metodologia sta avendo sempre più diffusione nella ricerca scientifica e promette di avere conseguenze rivoluzionarie soprattutto per le scienze dell'uomo, in forza dalle peculiarità del loro oggetto di studio.

Grazie alla simulazione è possibile ricreare in maniera artificiale anche fenomeni, quali quelli sociali ed economici, che non potrebbero essere studiati con altri strumenti. La possibilità di creare autentici laboratori sperimentali per l'analisi, rende possibile l'applicazione dei metodi della ricerca scientifica anche a scienze quali l'economia e la sociologia, molto spesso limitate al solo aspetto descrittivo delle loro teorie. Adottando il nuovo metodo, la teoria è incorporata nella simulazione. La capacità della simulazione di ricreare i fenomeni del reale verifica le ipotesi implicite nella teoria. In questo modo vi sarà una stretta interazione tra i fatti osservati e le formulazioni teoriche.

All'interno del secondo capitolo è dedicata particolare attenzione alla metodologia di simulazione con modelli basati su agenti. Adottando questo tipo di modelli è possibile creare mondi virtuali, popolati da molteplici ed eterogenei agenti artificiali, ognuno con proprie caratteristiche e propri schemi di azione, in grado di riprodurre fenomeni e strutture del mondo reale. Con questo strumento possono essere riprodotti non solo i sistemi semplici, a lungo oggetto di studio della scienza tradizionale a causa dei limiti cognitivi della mente umana, ma anche, e soprattutto, quelli complessi, caratterizzati da relazioni di tipo non lineare tra le parti. La complessità del sistema è il riflesso non solo del grado più o meno elevato di sofisticazione, introdotto nei singoli agenti, ma anche dell'interazione di quelli più semplici.

A questo riguardo, una specifica sezione del capitolo è dedicata all'analisi delle metodologie usate per la costruzione di questo tipo di modelli. In particolare, costituiscono oggetto di analisi le tecniche per la costruzione degli agenti, caratterizzati, a seconda delle scelte compiute, da diversi gradi di capacità cognitiva.

Il capitolo successivo, il terzo, analizza caratteristiche e proprietà degli strumenti e del linguaggio informatici usati per la costruzione di modelli simulativi. Sono descritte le potenzialità delle tecniche della programmazione ad oggetti, particolarmente adatta a rappresentare modelli e simulazioni ad agenti. Inoltre viene proposto uno specifico ambiente in cui sviluppare le applicazioni sperimentali: Swarm, linguaggio capace di gestire con semplicità le diverse componenti di un modello e la scansione degli eventi di una simulazione.

Nel quarto capitolo, sulla base degli strumenti e delle impostazioni metodologiche descritte in precedenza, si analizzano due modelli , l'*Artificial Stock Market*(ASM) e il *Genoa Market* che, partendo da assunzioni diverse, forniscono esempi di simulazioni applicate ai mercati azionari. Il terzo modello, *Surprising (Un)realistic Market* (SUM) sarà oggetto del capitolo successivo in virtù delle caratteristiche peculiari che lo distinguono dai due citati in precedenza. Il principale carattere distintivo tra SUM, da un lato, e ASM e *Genoa Market*, dall'altro, è rappresentato dal meccanismo di formazione del prezzo delle attività scambiate: nel modello SUM la formazione del prezzo di scambio avviene transazione dopo transazione, sulla base del confronto delle proposte di segno opposto immesse nel sistema dagli operatori, mentre negli altri due modelli il valore attribuito dal mercato alle attività è frutto del confronto di domanda e offerta aggregata, espressa dai singoli agenti.

SUM consente lo studio delle microfondazioni del mercato azionario, riproducendone il funzionamento. Il meccanismo di formazione del prezzo non adotta l'ipotesi di meccanismi di coordinamento esterno in grado di adeguare domanda e offerta aggregata. La dinamica del corso dell'attività trattata sul mercato è frutto dell'abbinamento delle proposte casuali espresse dai diversi agenti effettuata dal *book* del mercato. La capacità del modello descritto di generare in maniera spontanea bolle e *crash* ha rappresentato un primo risultato della simulazione, progressivamente resa più complessa e realistica.

La volontà di aprire il mondo di SUM, interamente popolato da agenti artificiali, all'interazione con "agenti umani", ha favorito l'estensione del modello verso l'esterno con la versione *SUM Web economic behaviour* (SUM-Web). L'obiettivo di questa versione è la costruzione di un autentico laboratorio sperimentale con cui effettuare l'analisi e lo studio dei fenomeni economici e dei comportamenti umani all'interno di un ambiente artificiale.

A livello di singolo operatore, i progressi nell'analisi delle motivazioni, non soltanto economiche, e nello studio dell'emergere di comportamenti irrazionali od imitativi potranno permettere una maggiore comprensione dei processi decisionali. A livello aggregato, i comportamenti complessi saranno frutto dell'interazione di soggetti che presentano modelli di comportamento così diversificati.

Indice

Capitolo 1

I mercati efficienti

> We must look at the price system as a . . . mechanism for communicating information if we want understand its real function. (Hayek, 1945)

1.1 Introduzione

L'ipotesi base su cui si sono fondate tutte le teorie finanziarie ortodosse degli ultimi decenni è quella del mercato efficiente. Il tema dell'efficienza informativa nei mercati risale agli albori della letteratura finanziaria con notevoli risvolti di natura empirica e modellistica. Per contro, una analisi microeconomica sull'argomento, presente in Grossman (1989), è più recente e si basa sui contributi della teoria delle aspettative razionali.

L'ipotesi di efficienza dei mercati parte dall'assunto secondo cui: se un insieme di informazioni non permette di prevedere le qotazioni future delle attività finanziarie, allora i loro prezzi riflettono le informazioni contenute in quel insieme informativo. In questo caso il mercato si dirà *efficiente* per quel set di informazioni.

Il valore di un'azienda, espresso dalle sue azioni, muta nel tempo in relazione alle prospettive di profitto per l'evolversi delle condizioni macroeconomiche, settoriali e di impresa. Il cambiamento delle condizioni e del valore si riflettono nel prezzo dell'azione se e quando le cause che hanno inciso sul mutamento dell'azienda sono portate a conoscenza del mercato.

Gli investitori valutano l'informazione e decidono se acquistare o vendere il titolo, con l'effetto di influenzare il prezzo della quota spingendolo ad allinearsi al suo valore intrinseco.

L'adeguamento del prezzo in risposta alle nuove condizioni dell'azienda può svilupparsi con vari gradi di tempestività rispetto alla diffusione dell'informazione. Se il prezzo risponde in maniera rapida e coerente al mutare delle situazioni, il mercato sul quale viene trattata l'attività finanziaria è definito efficiente. La rapidità dell'aggiustamento comporterà in media una

corrispondenza tra valore intrinseco della società e prezzo dei titoli quotati. Non tutti gli studiosi concordano con Fama (1970) nel richiedere alla definizione di mercato efficiente l'ipotesi che nei prezzi sia incorporato, in ogni momento, l'intero set di informazioni, garantendo in ogni istante la perfetta corrispondenza tra valore e quotazione: alcuni autori, tra cui Makiel (1992), definiscono mercato efficiente un mercato in cui vi sia l'impossibilità di realizzare extra-profitti negoziando in base alle informazioni disponibili. L'adozione di una simile ipotesi implica che le correzioni in grado di riportare il prezzo verso il valore intrinseco devono avere una rapidità tale da impedire la possibilità di ottenere extra-rendimenti al netto dei costi di transazione.

Un mercato per potere essere definito efficiente deve rispettare le condizioni seguenti:

- deve essere formato da una pluralità di operatori razionali, alla ricerca del massimo profitto, che agiscono in maniera non collegata;
- non vi sono costi di acquisizione dell'informazione, disponibile per gli agenti economici in ogni istante di tempo;
- le aspettative degli investitori sono omogenee: tutti gli investitori sono concordi nel valutare gli effetti delle informazioni sul prezzo delle attività e sulla loro dinamica;
- non è prevista la presenza di costi di transazione e di imposte.

Un simile schema teorico presenta requisiti difficilmente ritrovabili nella realtà in cui:

- il comportamento degli investitori non può mai dirsi completamente razionale, poiché affetto da illusioni cognitive ed emozioni che si traducono a livello collettivo in comportamenti di tipo imitativo;
- l'informazione non solo non è disponibile per tutti gli investitori simultaneamente ma spesso non è neppure gratuita; inoltre sui mercati vi sono fenomeni di *insider trading*;
- non è plausibile ritenere che le aspettative degli investitori siano omogenee: le informazioni possono essere interpretate in modo dissimile generando comportamenti eterogenei; inoltre, gli investitori, soprattutto quelli non professionisti, molto spesso possono reagire ad informazioni non rilevanti, tenendo conto di rumori ed indiscrezioni (nella letteratura anglosassone sono definiti *"noise investor"*);
- non è possibile prescindere dalla componente legata a costi di transazione ed imposizione fiscale.

Alla luce di questi riscontri empirici, le eccessive semplificazioni introdotte dalla teoria sono state parzialmente corrette, creando accezioni più deboli al paradigma proposto dall'ipotesi di mercato efficiente in grado di tener conto di questi elementi.

In linea con questi sviluppi non si richiede al mercato per essere efficiente che queste ipotesi siano perfettamente valide, ma soltanto in modo approssimato. Queste condizioni diventano sufficienti ma non necessarie perché il prezzo possa riflettere in maniera rapida e corretta le informazioni disponibili. Per l'efficienza del mercato sarà sufficiente che:

- sia nullo l'effetto delle azioni sui prezzi degli investitori con comportamento irrazionale, perché casuale o eliminato dall'intervento di altri operatori che compiono operazioni di arbitraggio riportando il prezzo verso quello corretto;
- la eterogeneità delle aspettative non consenta ad investitori di ottenere sistematicamente risultati migliori di quelli del mercato;
- vi sia un adeguato numero di investitori che abbia accesso alla notizia;
- gli effetti connessi con l'esistenza di costi di transazione e l'impatto della componente fiscale non siano tali da scoraggiare gli scambi.

Accentando queste ipotesi, raccolte sotto l'abbreviazione EMH[1], si hanno importanti conseguenze. In particolare, se nei prezzi è riflessa in maniera completa e corretta tutta l'informazione disponibile, le quotazioni rappresentano la miglior stima del valore intrinseco dell'attività e non esisteranno titoli sovra o sotto quotati. Da questa conseguenza discende che è impossibile per gli investitori ottenere in modo sistematico extra-profitti dall'attività di compravendita.

L'ipotesi su cui si basa la teoria nega la validità delle tecniche e degli strumenti di analisi correntemente utilizzati dagli operatori: l'*analisi fondamentale* e quella *tecnica*.

L'analisi fondamentale presuppone la possibilità di inefficienza del mercato nel valutare correttamente il valore intrinseco di una attività: con i propri strumenti analitici ricerca titoli sovra o sotto quotati da acquistare o vendere per ottenere extra-rendimenti. Ciò in perfetto contrasto con l'impossibilità, ipotizzata dalla EMH, di battere sistematicamente il mercato.

L'adeguamento immediato dei prezzi, invece, va contro uno dei principi alla base dell'analisi tecnica: la possibilità di individuare dei *trend* nelle quotazioni dei titoli. Per gli analisti tecnici l'adeguamento del prezzo al valore intrinseco non è immediato ma richiede tempo. Proprio la lentezza dell'adeguamento consente a chi utilizza questa metodologia operativa di individuare il *timing* con cui acquistare o vendere un titolo ottenendo degli

[1] Efficient Market Hipothesis.

extra-rendimenti. Anche qui si è in perfetto contrasto con la EMH che reputa impossibile battere in maniera continua il mercato.

1.2 EMH e il processo di formazione dei prezzi

L'ipotesi di mercato efficiente implica l'impossibilità a realizzare profitti speculativi in maniera sistematica. Accettando il modello di gestione del portafoglio proposto dal CAPM, questa conclusione si traduce, a livello gestionale, nel risultato di ottenere da ogni investimento un rendimento coerente con il relativo profilo di rischio. Nello spazio cartesiano rendimento atteso-volatilità dell'attività, tutti i titoli, correttamente valutati, si collocheranno perfettamente lungo la retta della *Security Market Line*, la SML.

In un mercato non efficiente, invece, i titoli tenderanno a disperdersi intorno alla SML, con alcune attività sovra ed altre sottoquotate. La dispersione di questi titoli consente agli investitori di adottare, ottenendo extra-profitti, strategie di arbitraggio che riportano i titoli versi i valori corretti. L'EMH ammette tale possibilità, nell'ipotesi che queste situazioni non si verifichino in maniera sistematica.

Se il prezzo riflette l'intero set informativo con rapidità e correttezza allora non è possibile scegliere investimenti che realizzino risultati migliori di quelli di equilibrio.

Le informazioni disponibili al tempo t incorporate nei prezzi dei titoli sono già scontate e il rendimento atteso per il periodo $t+1$, dato il set di informazioni Z_t, è uguale a quello che si stima ignorando Z_t. Applicando il CAPM, questa conclusione implica che Z_t non indica all'investitore un rendimento atteso diverso da quello normalmente suggerito dalla SML.

In questo modo è indicata anche la strategia di gestione del portafoglio da seguire. Sarà sufficiente scegliere i titoli in modo da ottenere la combinazione desiderata tra rischio e rendimento tale da ridurre al minimo le transazioni ed i relativi costi. Solo chi non accetta l'ipotesi cercherà di adottare strategie di investimento diverse da quella ottima.

L'EMH può essere così formalizzata:

$$Z^*_{t-1} = Z_{t-1} \tag{1.1}$$

$$f^*(\frac{P_t}{Z^*_{t-1}}) = f(\frac{P_t}{Z_{t-1}}) \tag{1.2}$$

dove:

- Z_{t-1} è l'insieme di informazioni disponibili al tempo $t-1$, utilizzato per la determinazione del prezzo del titolo al tempo $t-1$ e del valore atteso in t;
- Z^*_{t-1} è il set di informazioni effettivamente utilizzato dagli operatori;

- $f^*(\ldots)$ e $f(\ldots)$ sono le funzioni di distribuzione di probabilità condizionata della quotazione dell'attività al tempo t.

Con la [1.1] si descrive in maniera rigorosa che tutto il set di informazione disponibile in un dato momento temporale è utilizzato dagli operatori del mercato in maniera istantanea. Mentre la [1.2] conferma che la funzione $f^*(\ldots)$ tiene conto del complesso delle informazioni esistenti all'istante $t-1$ ed implica che:

$$E(\frac{P_t}{Z^*_{t-1}}) = E(\frac{P_t}{Z_{t-1}})$$

dove E è l'operatore di valore atteso. Lo sviluppo del processo di formazione del prezzo all'istante $t-1$ può essere così sintetizzato:

- le informazioni Z_{t-1} sono liberamente disponibili ed utilizzate dagli operatori secondo la [1.1] per formulare le previsioni sulla distribuzione di probabilità $f^*(\frac{P_t}{Z^*_{t-1}})$ e il valore atteso $E(\frac{P_t}{Z^*_{t-1}})$ dei prezzi al tempo t;

- formulate le aspettative, il mercato determina il rendimento atteso di equilibrio $E(\frac{R_t}{Z^*_{t-1}})$;

- accettando l'ipotesi implicita nella EMH che l'investitore agisca attendendosi un rendimento positivo maggiore per gli investimenti più rischiosi sulla base di un opportuno premio per il rischio, sarà possibile definire il rendimento atteso di un periodo come:

  $$E(\frac{R_t}{Z^*_{t-1}}) = \frac{E[(\frac{P_t}{Z^*_{t-1}})]}{P_{t-i}}$$

  Non viene escluso che il rendimento atteso possa anche essere negativo. Neppure si richiede che rendimento atteso ed effettivo debbano essere uguali: i due rendimenti possono essere differenti, ma in un mercato efficiente in cui i prezzi costituiscono la migliore stima del valore intrinseco del titolo, la loro differenza attesa sarà nulla. Il prezzo atteso sarà definito come:

  $$E(\frac{P_t}{Z^*_{t-1}}) = [1 + E(\frac{R_t}{Z^*_{t-1}})]P_{t-1}$$

  tutto ciò premesso, la quotazione del titolo al tempo $t-1$ sarà:

  $$P_{t-1} = \frac{E(\frac{P_t}{Z^*_{t-1}})}{[1 + E(\frac{R_t}{Z^*_{t-1}})]}$$

  con prezzo di equilibrio pari a

  $$P_{t-1} = P(e)_{t-1}$$

Se P_{t-1} non è coerente con prezzo e rendimento attesi, ciò si tralucerà in una variazione della quotazione in $t-1$.

Ipotizzato che, per un dato livello di prezzo e rendimento attesi, sia $Pt-1 < P(e)_{t-1}$ allora il prezzo nello stesso istante sarà destinato a crescere per riportarsi al valore corretto, grazie ad un flusso di acquisti istantaneo. Un simile meccanismo di adeguamento rende impossibile per gli operatori ottenere extra-rendimenti.

L'ipotesi di efficienza del mercato implica che questo aggiustamento del prezzo avvenga istantaneamente dal momento che le informazioni sono disponibili al medesimo istante e sono interpretate correttamente da tutti gli operatori. Questa rapidità della correzione rende impossibile realizzare extra-profitti operando sul titolo al tempo $t-1$ sulla base della conoscenza delle informazioni Z_{t-1}.

Essendo questo flusso di input liberamente disponibile, allora al tempo t il prezzo di mercato P_t non differirà da quello atteso $E(\frac{P_t}{Z_{t-1}})$:

$$E(\frac{P_t}{Z^*_{t-1}}) - P_t = f(\frac{u_t}{Z_{t-1}})$$

$$E(\frac{u_t}{Z_{t-1}}) = 0$$

con u_t che rappresenta l'errore di previsione.

L'ipotesi di efficienza del mercato porta a ritenere che l'errore u_t sia in media pari a zero e questo implica che la previsione di P_t sia corretta: il prezzo atteso è in media uguale a quello effettivo.

$$P_t = E(P_t) + u_t$$

$$E[\frac{E(P_t)}{Z_{t-1}} - P_t] = E(P_t) - E(P_t) = 0$$

con $E(u_t) = 0$.

1.3 Il modello dei prezzi basato sul *random walk*

Nell'ambito delle ipotesi di efficienza dei mercati si è distinto un particolare modello di formazione dei prezzi basato sull'ipotesi del *random walk.* Sono tre le versioni del modello che, partendo sostanzialmente dagli stessi assunti, dipendono da diverse distribuzioni delle variabili stocastiche.

Il modello poggia sull'assunto che la variazione di una variabile stocastica P_t che deriva da una distribuzione identica[2] con media pari a zero è

[2]Una variabile casuale è identicamente distribuita se:
$E(u_t) = 0$; $E(u_t)^2 = \sigma^2$;
e
$E(u_{t-s}) = 0$; $E(u_{t-s})^2 = \sigma^2$
ossia se ciascuna variabile ha lo stesso tipo di distribuzione di probabilità con gli stessi parametri distributivi: identica media e varianza.

indipendente dalle precedenti:

$$P_t = P_{t-1} + u_t \tag{1.3}$$

$$\delta_t = u_t$$

dove u_t è una variabile casuale indipendente e identicamente distribuita; da cui si ha che $E(u_t) = 0$, $E(u_t^2) \neq 0$, la varianza è pari a $E(u_t)^2 = \sigma^2$ e l'autocovarianza è nulla $E(u_t, u_{t-s}) = 0$. Il valore atteso della variabile stocastica che segue un processo del tipo *random walk* è una costante, mentre la varianza del processo è dipendente dal tempo. Dato che la varianza non è costante, il processo non è stazionario nel tempo. Sulla base di questa ipotesi:

$$E(P_t) = P_{t-1}$$

da cui segue che:

$$P_t = E(P_t) + u \tag{1.4}$$

$$E(\delta P_t) = 0$$

La [1.4] esprime in maniera formale l'assunto che il prezzo ad ogni istante sia uguale a quello al tempo precedente eventualmente corretto da un errore stocastico. Se si suppone di avere la storia passata dei prezzi nei vari istanti, da 0 a t, indicata con $P_0, \ldots, P_t$, la stima di P_{t+1} sarà uguale al valore atteso della distribuzione di P_{t+1} condizionata[3] della serie storica dei prezzi precedenti:

$$\hat{P}_{t+1} = E(\frac{P_{t+1}}{P_0, \ldots, P_t})$$

Avendo ipotizzato $\hat{P}_{t+1}$ indipendente da $P_0, \ldots, P_t$, la stima di prezzo per $t+1$ sarà data da:

$$\hat{P}_{t+1} = E(P_{t+1})$$

e dato che $E(P_{t+1} = P_t$ allora:

$$\hat{P_{t+1}} = P_t$$

Questo risultato implica che, per l'ipotesi del *random walk*, la miglior stima del prezzo futuro è semplicemente il prezzo attuale.

Possiamo sintetizzare il modello con le due uguaglianze seguenti:

$$f(\frac{R_t}{Z_{t-1}}) = f(R_t) \text{ dove } Z_{t-1} = f(R_{t-1}, \ldots, R_1)$$

$$f(R_t) = f(R_{t-1}) = f(R_{t-n})$$

Le due uguaglianze esprimono l'indipendenza della funzione di distribuzione dei rendimenti e l'identità della stessa nel tempo.

[3]Inoltre sotto le ipotesi citate le probabilità condizionate e marginali sono uguali.

L'ipotesi di efficienza del mercato non presuppone che rendimenti e prezzi siano perfettamente indipendenti. Gli assunti alla base della teoria presuppongono che le informazioni siano pienamente riflesse nelle quotazioni e che il prezzo del giorno successivo sia una variabile casuale dipendente dalle notizie attuali. La possibilità che prezzi e rendimenti siano debolmente dipendenti non modifica l'impostazione teorica: le due variabili sono considerate indipendenti se la correlazione è modesta in valore assoluto e tale da non consentire extra-profitti significativi e sistematici agli operatori che cercano di approfittare della relazione. Prescindendo dall'ipotesi di indipendenza di prezzo e rendimento, il cardine del modello diventa la verifica dell'altra ipotesi: quella dell'andamento casuale delle due variabili. Le caratteristiche del mercato tornano ad essere determinanti per consentire alla relazione di essere verificata.

Per contro, secondo l'ipotesi del *random walk* non solo il valore atteso della futura variazione dei prezzi è indipendente dalla precedente, ma anche l'intera distribuzione di probabilità è costante nel tempo.

Il modello non specifica la forma della distribuzione; gli unici requisiti che questa deve possedere sono rappresentati dalla stazionarietà dei parametri, che presuppone la stabilità della distribuzione e l'identità della variabile casuale.

Gli studi basati su dati giornalieri e mensili riferiti al mercato azionario americano compiuti da Fama (1970) hanno dimostrato come:

- la curva delle variazioni giornaliere dei rendimenti è leptocurtica;
- la distribuzione dei rendimenti mensili, invece, è approssimabile con una normale.

Lo studioso ha poi proposto una distinzione all'interno della tesi generale dell'efficienza del mercato individuando tre sotto-ipotesi sulla base delle informazioni disponibili.

- *Weak Hypothesis*: la forma debole secondo cui l'insieme informativo include la serie storica dei prezzi e dei rendimenti dei titoli;
- *Semi Strong Hypothesis*: la forma semiforte, in linea con la quale l'insieme informativo include le informazioni di dominio pubblico;
- *Strong Hypotesis*: la forma forte, secondo cui l'insieme informativo include le informazioni di dominio pubblico ed ogni informazione privata. Si divide a sua volta in due accezioni: efficienza quasi forte ed efficienza forte in senso stretto.

Tramite la legge della speranza condizionata è stato verificato che l'efficienza forte per un mercato implica l'efficienza in forma semiforte che a sua volta implica quella in forma debole.

La verifica dell'efficienza di un mercato consiste nell'individuare il grado di prevedibilità dei prezzi della attività oggetto di negoziazione; dalla capacità di previsione deriverà o meno quella di battere i rendimenti del mercato, avvalorando progressivamente le tesi proposte.

In letteratura, le ipotesi di efficienza debole e semiforte hanno trovato parziale verifica empirica con le assunzioni di non arbitraggio sui prezzi-rendimenti dei titoli. La letteratura stessa, dimostrando come i rendimenti dei titoli possano essere previsti, ha dato esiti negativi nella verifica empirica dell'efficienza in forma forte.

1.4 L'ipotesi di efficienza in forma debole

La forma debole, *Weak Hypotesis*, parte dall'assunto che i prezzi delle attività incorporano tutte le informazioni che possono essere tratte dal mercato e dai suoi dati.

Per fare un esempio, un andamento stagionale dei prezzi passati di una attività è conosciuto dal mercato che tenderà a trasferire nelle quotazioni tale informazione, eliminando il fenomeno. Ciò nega la validità del metodo operativo basato sull'analisi tecnica, che con questa ipotesi viene ad essere privata di uno dei suoi principi chiave.

L'efficienza di mercato indicata da quest'ipotesi è il riflesso della capacità dei prezzi di incorporare tutte le informazioni passate. Conoscere il passato non aiuta gli operatori ad ottenere rendimenti superiori al mercato.

Riprendendo le formalizzazioni dei paragrafi precedenti, il modello può essere così sintetizzato:

$$Z_{t-1} = Z^*_{t-1}$$

$$E(\frac{R_t}{Z_{t-1}}) = E(\frac{R_t}{Z^*_{t-1}})$$

dove con Z_{t-1} si indica la serie storica dei prezzi, dei rendimenti e dei volumi scambiati.

Nella forma debole, le informazioni disponibili e gratuite sono utilizzate dagli investitori (Z^*_{t-1}) grazie ad aspettative omogenee per operare in un mercato che non presenta costi di transazione. Queste condizioni stringenti possono essere rese più flessibili senza far perdere al mercato il requisito dell'efficienza secondo questa accezione: sarà sufficiente riferirsi a mercati in cui le notizie sono disponibili per un gran numero di investitori, in cui non vi siano investitori capaci di interpretare meglio degli altri gli input in maniera sistematica e in cui non vi sono costi transattivi capaci di inibire gli scambi.

La verifica di queste condizioni implica che i prezzi incorporino rapidamente e completamente l'informazione. In questo caso, il prezzo sarà la migliore stima del valore intrinseco e la correzione istantanea dei prezzi impedirà di sfruttare i processi di adeguamento del prezzo al valore.

Per la verifica del modello occorre ipotizzare che i rendimenti attesi[4] siano costanti nel tempo e siano legati dalla relazione seguente:

$$E(\frac{R_t}{Z_{t-1}}) = E(R_t) = E(R)$$

con

$$E(R_t) = [\frac{E(\frac{R_t}{Z_{t-1}}) - P_{t-1}}{P_{t-1}}] = E(R)$$

Assumendo che i prezzi in ogni intervallo temporale possano essere rappresentati in funzione due variabili casuali, il prezzo precedente e la variazione di periodo, dalla equazione seguente:

$$P_t = P_{t-1} + \delta P$$

dove δ si ipotizza rappresentabile da:

$$\delta = aP_{t-1} + bP_{t-1}v \text{ con } E(v) = 0$$

dove a e b sono costanti e v è una variabile casuale con media nulla. La media nulla di v permette di scrivere:

$$E(P_t) = P_{t-1} + aP_{t-1}$$

Essendo il rendimento definibile come:

$$R_t = \frac{P_t - P_{t-1}}{P_{t-1}}$$

ed essendo P_t definibile come:

$$P_t = P_{t-1} + aP_{t-1} + bP_{t-1}v$$

è possibile scrivere:

$$R_t = a + bv$$

Dato che $E(v) = 0$, allora il rendimento atteso è costante e non dipende da R_{t-q} con $q = 1, \ldots, n$:

$$E(R_t) = a$$

La conclusione è che $E(R_{t+1})$ non dipende da R_t, cioè dal passato. Se, invece, $E(R_{t+1})$ non è costante è possibile asserire che $E(R_{t+1})$ è correlato con R_t e quindi l'evoluzione dei prezzi dipende dal passato: il mercato non è efficiente.

[4]Per la verifica del modello si utilizzano i rendimenti in forza della loro distribuzione di frequenza stabile contrariamente a quella dei prezzi che cambia nel tempo.

Rendimenti costanti consentono di affermare che la retta di regressione di R_{t+1} su R_t è pari alla costante E. Ipotizzata una relazione lineare in R_t:

$$E(\frac{R_{t+1}}{R_t}) = a + bR_t$$

affinché $E(\frac{R_{t+1}}{R_t})$ non dipenda da R_t, il valore di b deve essere pari a zero per tutti i ritardi sottoposti a verifica.

La verifica del modello avviene quindi con il test di correlazione con cui è possibile stabilire se le variazioni dei prezzi sono linearmente dipendenti nel tempo, esaminando il valore del coefficiente delle variazioni dei prezzi successivi.

L'applicazione del test al fine della verifica del modello per il mercato italiano (su 30 titoli dal 1978 al 1983) in Caparrelli (1986) ha fornito una serie di riscontri interessanti. Il primo dei quali dimostra come la rappresentatività del modello migliori allungando l'intervallo dei ritardi. L'ipotesi appare confermata per ritardi mensili, che consentono al mercato di assorbire nel prezzo l'informazione. Dal confronto tra uno studio analogo compiuto da Fama sui titoli americani con quello di Caparrelli, è risultato che il mercato italiano è meno efficiente rispetto a quello d'Oltreoceano.

Recenti studi (SIAT, 2002) hanno evidenziato un progressivo aumento dell'efficienza dei mercati negli anni novanta grazie alle innovazioni tecnologiche e normative che hanno reso i listini mondiali sempre più interdipendenti. La progressiva introduzione dell'informatica nei sistemi di contrattazione da un lato ha favorito la funzionalità degli scambi e dall'altro ha comportato una significativa riduzione dei costi di transazione. Inoltre l'informatica e le tecnologie multimediali hanno reso più veloce la trasmissione delle informazioni, rendendo più rapida la capacità di reazione degli operatori alle notizie. Ciò ha verificato un aumento della correlazione tra le aree geografiche ed una maggior capacità di risposta agli input da parte dei mercati, superando in parte le conclusioni precedenti.

1.5 L'ipotesi di efficienza in forma semiforte

La seconda ipotesi, *Semi Strong hypothesis*, tiene conto non solo delle informazioni passate, ma di tutta l'informativa pubblica nel processo di formazione del prezzo delle attività. In un mercato efficiente secondo questa ipotesi è negata la possibilità di ottenere extra-rendimenti sulla base delle informazioni pubbliche, già scontate nei prezzi attuali. Ad essere negata è la validità della analisi fondamentale.

Nella forma semiforte di efficienza, le informazioni aziendali ed economiche sono riflesse dai prezzi delle attività in maniera rapida e corretta. Il set informativo che caratterizza questa ipotesi è più ampio di quello che contraddistingue la forma debole: comprende tutte le notizie pub-

bliche, derivate da fonti non riservate, che riguardano dati e notizie frutto dell'analisi dell'informativa contabile rilasciata dagli emittenti.

Un modello di verifica è proposto da Caparrelli (1989). Per l'analisi dell'ipotesi è necessario isolare l'effetto del fattore specifico (rappresentato dalla informazione pubblica attinente l'impresa) dal macro fattore (dovuto alle cause generali dell'ambiente). Per distinguere l'impatto dei due fattori si è reso necessario ricorrere alla costruzione di un indice singolo che scomponga le componenti del rendimento.:

$$R_{it} = \alpha_i + \beta_i R_{mt} + e_{it}$$

dove α_i è il rendimento medio del periodo; β_i è il coefficiente di reattività di R_i alla influenza dei fattori macroeconomici; R_m è il rendimento dell'indice M; e_i è la quota di R_i rappresentata dal fattore specifico.

L'equazione distingue i fattori che determinano il rendimento, R_{it}, tenendo conto di tutte le informazioni disponibili al tempo t. Grazie al valore atteso dell'equazione è possibile identificare l'effetto dovuto al macro fattore riassunto in R_{mt}:

$$E(R_i) = \alpha_i + \beta_i E(R_{mt})$$

R_{mt} è ignoto al tempo $t-i$: di questo rendimento è nota solo la distribuzione di probabilità $f(\frac{R_{mt}}{Z_{t-1}})$ che dipende da quella di tutti i titoli al tempo t ossia $f(P_1, \ldots, P_{nt})$ e dalle quotazioni effettive a $t-1$ determinate sulle informazioni Z_{t-1}.

Esplicitando la relazione in funzione del set di informazioni disponibile all'istante $t-1$ si ottiene:

$$E(\frac{R_{it}}{Z_{t-1}}) = \alpha_i + \beta_i E(\frac{R_{mt}}{Z_{t-1}})$$

che, per verificare l'ipotesi, deve coincidere con R_{mt} al fine di escludere ogni influsso del macro fattore sul rendimento; ciò tenendo conto che il rendimento in eccesso positivo o negativo $(\frac{e_{it}}{Z_{t-1}})$ al tempo t dovuto alla sola informazione aziendale è uguale alla:

$$R_{it} - E(\frac{R_{it}}{Z_{t-1}}) = \frac{e_{it}}{Z_{t-1}}$$

In fase di verifica *ex-post* dell'ipotesi si assume che la condizione necessaria:

$$E(\frac{R_{it}}{Z_{t-1}}) = R_{mt}$$

sia garantita calcolando il rendimento con distribuzione normale in funzione di R_{mt}.

Nell'ipotesi che all'istante t non si renda disponibile un nuovo set informativo (Z_t)[5], dalle equazioni precedenti deriva che:

$$E(\frac{e_{it}}{Z_{t-1}}) = 0$$

Da queste conclusioni risulta evidente che per verificare l'ipotesi di efficienza semiforte sarà sufficiente l'analisi del comportamento del residuo. e_{it} prima e dopo la diffusione di una eventuale notizia deve essere uguale a zero, allontanandosi da questo valore solo nel momento in cui l'informazione diviene pubblica.

Fama ed altri hanno suggerito un metodo per la verifica di questa ipotesi che si basa sulle seguenti fasi:

- occorre stimare il modello di mercato per un periodo antecedente e seguente l'intervallo temporale in cui si vuole esaminare la dinamica del residuo;
- dalla stima precedente si ottengono i valori corretti di α_i e β_i che assieme al rendimento R_m permettono di valutare il rendimento normale:
$$\hat{R_{it}} = \hat{\alpha_i} + \hat{\beta_i} R_{mt}$$
- successivamente si determina il residuo giornaliero come scarto tra il rendimento effettivo R_{it} e quello normale $\hat{R_{it}}$.

Il metodo è stato applicato dagli stessi autori e da altri studiosi per la verifica della ipotesi di efficienza semiforte. Per valutare l'impatto dell'informazione sui prezzi sono stati analizzati eventi societari quali l'assegnazione gratuita di nuove azioni o il frazionamento delle stesse (*split*), la comunicazione delle politiche dei dividendi e la diffusione dei consigli degli analisti.

La ricerca di conferme all'ipotesi ha reso necessario la distinzione tra i casi in cui:

- prezzo e rendimento non cambiano per effetto della nuova informazione;
- prezzo e rendimento si modificano sulla base delle nuove informazioni; ciò verifica l'ipotesi di efficienza semiforte e determina tre scenari ulteriori:
    a) l'informazione non è anticipata dal mercato: in questo caso i rendimenti saranno normali prima dell'evento e successivamente l'impatto della notizia modificherà il prezzo dell'attività, consentendo agli operatori di ottenere extra-rendimenti fino a quando sarà raggiunto il valore corretto;

[5]Nel caso opposto, R_{it} dipenderà da Z_t e $E(\frac{e_{it}}{Z_{t-1}}) \neq 0$

b) la notizia è già scontata: in questa ipotesi il prezzo dell'attività si adegua in maniera graduale e progressiva prima della comunicazione della notizia; al momento della comunicazione non vi sarà alcuna reazione del prezzo che tenderà a stabilizzarsi insieme ai rendimenti; il titolo sarà nuovamente correttamente valutato;

c) prescindendo dall'impatto della notizia, il mercato è inefficiente e non riporta il titolo al valore corretto, non riuscendo a valutare gli effetti dell'informazione.

Le analisi compiute nel caso di distribuzione di nuove azioni e di *split* hanno verificato solo in parte l'ipotesi semiforte. Il raggiungimento del prezzo di equilibrio è avvenuto in molti casi solo dopo una reazione in eccesso iniziale a cui è seguita una correzione con un ritardo anche piuttosto lungo. Questo riscontro indebolisce l'ipotesi di efficienza semiforte che presuppone che l'informazione pubblica sia trasferita nella quotazione di equilibrio in maniera rapida e corretta.

Lo studio delle serie storiche dei titoli in occasione dell'annuncio dei dividendi da parte delle società emittenti ha dato maggiori indicazioni di verifica alla tesi di efficienza semiforte. Nei periodi precedenti l'evento, il prezzo ha mostrato una tendenza a muoversi in maniera correlata con il prossimo annuncio, mostrando una buona capacità di anticiparne gli effetti. Tale adeguamento preventivo potrebbe trovare giustificazione nella valutazione dell'andamento della gestione della società, basato su altre notizie, e dall'azione di operatori informati in precedenza. Questa capacità di previsione sembra aver dato maggiori risultati nell'anticipare eventi positivi mentre sembra non aver dato le stesse risposte in termini di efficienza di mercato nel caso di annunci negativi. Per società di dimensioni ridotte si è anche verificato come il fattore specifico abbia maggior impatto sui prezzi del titolo rispetto al caso dei grandi gruppi, più sensibili al macro fattore.

Nello studio dell'impatto dei giudizi degli analisti diffusi dai *mass media* è stato osservato come i prezzi abbiano mostrato la tendenza a muoversi prima che il consiglio diventasse pubblico. Inoltre si è osservata una maggiore reattività dei prezzi nel caso di consigli di acquisto rispetto al caso opposto. Il fenomeno è stato attribuito alle operazioni dei clienti degli analisti che hanno agito correttamente con il consiglio ricevuto prima della relativa pubblicazione. Questi clienti, inoltre, sembrano aver privilegiato le indicazioni di acquisto rispetto a quelle in vendita a tal punto da indurre gli stessi analisti ad assecondarli, limitando i propri giudizi negativi. Un altro aspetto interessante di queste ricerche ha riguardato la dimensione della società emittente i cui titoli erano oggetto di analisi. Le indicazioni si sono mostrate più efficaci se aventi oggetto società minori mentre non lo sono state per quelle più grandi e più capitalizzate. Ciò ha portato a concludere che il mercato ha un efficienza informativa diversa a seconda della dimensione della società

quotata: è più efficiente per quelle a grande capitalizzazione (*blue chips*) rispetto a quelle minori (*small caps*).

Gli adeguamenti dei prezzi non si sono rivelati istantanei, ma hanno consentito l'ottenimento di extra-profitti. Secondo una ricerca di Barber e Loeffler (1993) nel processo di adeguamento del prezzo è possibile distinguere due effetti: uno di breve ed uno di lungo periodo. Il primo è causato dall'azione degli investitori meno esperti che trasferiscono l'evento nei prezzi accogliendo il consiglio in maniera acritica; il secondo, invece, si propaga nel tempo e deriva dalla valutazione del consiglio da parte degli altri operatori che, riconoscendone la fondatezza, lo incorporano nei loro comportamenti, avvalorando l'attendibilità dell'esperto.

1.6 L'ipotesi di efficienza in forma forte

La forma forte di efficienza del mercato, *Strong Hypothesis*, ipotizza che tutti gli operatori dispongano dello stesso set di informazioni, escludendo la presenza di operatori in grado di accesso "privilegiato" alle notizie.

Secondo questa ipotesi l'informazione è disponibile a tutti gli operatori nello stesso momento. Ciò ha impatti sul modello di diffusione delle notizie che, nella struttura teorica proposta da Fama, non si basa sui cerchi concentrici, ma è "a pioggia".

La condizione sufficiente perché il mercato possa essere considerato efficiente nella forma forte è che l'insieme Z_t^* includa tutta l'informativa disponibile, compresa quella privata. Per informativa privata si intende il complesso delle notizie di cui possono essere in possesso, a qualunque titolo, particolari categorie di soggetti che sono connesse all'attività della società emittente.

Se il mercato, al tempo t, basa la propria valutazione su di un set informativo diverso da Z_t^*:

$$Z_{t-1}^* \neq Z_{t-1}$$

l'effetto delle nuove notizie si tradurrà in un prezzo stimato per il periodo successivo diverso da quello effettivo:

$$E(\frac{P_t}{Z_{t-1}^*}) \neq P_t$$

La stima corretta potrà essere compiuta solo da chi possiede l'informazione privata:

$$Z_{t-1}^* = Z_{t-1}$$

$$E(\frac{P_t}{Z_{t-1}}) = P_t$$

In linea con questo schema, l'operatore che, nella valutazione delle proprie stime utilizza tutta l'informazione Z_{t-1}, valuterà l'attività finanziaria ad

un prezzo diverso da quello di equilibrio ed agirà sulla base di queste informazioni supplementari. Una volta che queste saranno note, il prezzo si adeguerà a quello stimato dall'operatore informato che potrà chiudere la operazione concordemente con le proprie attese.

L'ipotesi di efficienza del mercato nega la possibilità sia che vi possano essere operatori caratterizzati dalla abilità di stimare prezzi e rendimenti sistematicamente corretti (nella sua accezione quasi forte) sia che il rendimento medio possa essere superato ricorrendo alla informativa privata (ipotesi di efficienza in senso stretto). Nessun operatore dovrebbe ottenere sistematicamente extra-profitti dalla propria attività.

La verifica dell'ipotesi non ha dato risultati completamente in linea gli assunti descritti. Molti studiosi, seguendo lo schema originale seguito da Jensen (1964), hanno analizzato i risultati ottenuti dai fondi comuni di investimento. Questi strumenti finanziari, essendo gestiti da operatori professionisti razionali e perfettamente informati, dovrebbero essere in grado di ottenere sistematicamente risultati superiori a quelli del portafoglio di mercato. Questa tesi viene negata dai lavori citati che indicano come un risparmiatore inesperto che avesse investito nell'attività priva di rischio e nel portafoglio di mercato, avrebbe ottenuto un rendimento superiore a quello medio dei fondi comuni. L'ipotesi di una maggior capacità dei gestori professionisti di ottenere rendimenti superiori viene negata e questo risultato è coerente con l'ipotesi di efficienza nella accezione quasi forte.

Per contro, la analisi dell'ipotesi forte appare decisamente più problematica in forza della necessità di verificare l'impossibilità degli operatori ad ottenere extra-profitti sebbene in caso di possesso di informazioni private.

1.7 Critiche al modello EMH e le anomalie del mercato

Negli ultimi anni, nonostante le testimonianze a sostegno della teoria siano aumentate, l'ipotesi di efficienza dei mercati sembra messa a dura prova da sfide sia teoriche che sperimentali che possono essere così raggruppate:

- critiche teoriche;
- anomalie ed inefficienze del mercato;
- evidenze empiriche e sperimentali sul reale comportamento degli investitori.

Solo il successo nel fornire risposte soddisfacenti in questi ambiti di analisi potrà permettere alla teoria proposta di continuare ad essere percepita come la miglior spiegazione di una realtà sempre più complessa.

La prima critica teorica riguarda l'ipotesi di comportamento razionale degli investitori nel processo di formazione del prezzo.

Evidenze empiriche e sperimentali hanno dimostrato che il comportamento degli operatori, affetto da illusioni cognitive e influenzato dalle emozioni, non può mai dirsi completamente razionale. A ciò si aggiunga che gli schemi di comportamento degli operatori, deviando dall'ipotesi di razionalità, non determinano azioni interamente casuali, ma sviluppano direzioni e tendenze nell'agire. Questo mette ulteriormente in crisi il modello, negando l'ipotesi di comportamenti casuali.

Tornando alla valutazione degli schemi di azione degli investitori anche quelli professionisti sono affetti dagli stessi problemi che affliggono quelli meno esperti, e possono dar vita a comportamenti irrazionali. Un esempio è la tendenza dei gestori a selezionare le stesse attività dei colleghi per non correre rischi di differenziarsi dalla massa.

Molto spesso le difficoltà di stima delle aspettative di lungo termine che determinano il prezzo di equilibrio del titolo spingono gli investitori a concentrare la loro attenzione sulle implicazioni di breve delle informazioni. Adottando questo schema, il prezzo corrente è la miglior stima del prezzo che gli operatori si attendono per il periodo futuro, e non la stima del valore attuale dei flussi futuri generati dall'attività finanziaria.

In questo senso, il mercato sarà efficiente solo per la capacità di trasferire rapidamente nei prezzi le informazioni che hanno effetti di breve periodo, mentre sarà inefficiente nel lungo periodo in forza dell'incapacità a valutare correttamente tutto il set di informazioni disponibile.

Inoltre, alle difficoltà cognitive del singolo si sommano spesso fenomeni di imitazione generale che danno vita ad illusioni collettive, trascendendo i limiti della psiche per diventare fenomeni sociali. L'effetto di queste azioni si traduce a livello di mercato in esagerate o carenti reazioni dei titoli all'informazione. Ricerche empiriche hanno mostrato come le reazioni carenti, che determinano il trasferimento lento dell'informazione nei prezzi, si estendono per periodi da uno a dodici mesi mentre le reazioni esagerate tendono a prolungarsi più lunghi, da tre a cinque anni. A rafforzare i comportamenti alla base delle reazioni esagerate intervengono continui e ripetuti messaggi di rinforzo che convincono gli operatori della razionalità delle loro scelte. Le fasi di allontanamento dei prezzi dai valori corretti sono seguite da profonde correzioni in grado di riportare i prezzi in corrispondenza dell'equilibrio.

Gli eccessi di reazione del prezzo delle attività erano già noti a Keynes cha aveva osservato come i corsi di borsa fossero eccessivamente influenzati dalle fluttuazioni del valore atteso dei profitti della società. Queste fluttuazioni avevano, secondo l'economista, la capacità di far reagire gli operatori a notizie positive (negative) favorendo una crescita (diminuzione) dei titoli oltre il loro valore intrinseco di equilibrio.

La teoria dei mercati finanziari ha tentato l'analisi e lo studio di questo fenomeno con l'ipotesi dell'eccesso di reazione (*Overreaction Hypothesis - (OH)*). Alla base dell'ipotesi, nella formulazione originale, vi sarebbe l'osservazione di due effetti distinti:

- l'elemento dovuto all'*effetto direzionale* secondo cui variazioni estreme dei prezzi sono seguite da movimenti di segno opposto. In linea con questa proposizione, titoli che hanno ottenuto un rendimento periodale elevato avrebbero alte probabilità di ottenere risultati negativi nel successivo e viceversa. La verifica di queste ipotesi si è avuta con l'opera di DeBondt e Thaler (1985) che per primi hanno dimostrato l'esistenza dell'effetto direzionale;

- l'elemento imputabile all'*effetto grandezza* secondo cui maggiore è la variazione iniziale del prezzo, più è forte la reazione che ne segue. Questo implica che se un titolo ha ottenuto il miglior rendimento periodale, nel successivo intervallo di tempo registrerà il peggiore e viceversa.

Altri studiosi, recentemente, avrebbero individuato, per definire l'ipotesi di eccesso di reazione, un terzo effetto, indicato come *effetto intensità*, secondo cui minore è la durata della variazione iniziale del prezzo, più forte è la risposta successiva.

Le verifiche dell'ipotesi dell'eccesso di reazione hanno confermato le ipotesi dei tre effetti nel breve periodo (con ottica periodale mensile). Allargando l'orizzonte temporale di valutazione, i risultati dei test compiuti hanno confermato l'esistenza dell'effetto intensità mentre l'esistenza dell'effetto grandezza sarebbe limitata agli eventi negativi. Il mercato, secondo queste conclusioni, avrebbe un comportamento asimmetrico, attribuendo maggior peso alle notizie negative.

Secondo alcuni studiosi, nel lungo periodo l'ipotesi di eccesso di reazione dei mercati non troverebbe conferme non risultando verificato l'effetto di direzione. L'ipotesi dell'eccesso di reazione, ovviamente, è negata da quella di efficienza dei mercati con cui è in netto contrasto.

La seconda critica mossa all'ipotesi verte sulla reale possibilità per gli arbitraggisti di riportare i prezzi al loro valore intrinseco, annullando gli effetti dei disturbi sui prezzi, frutto di comportamenti irrazionali. La critica trae forza dall'osservazione dei reali comportamenti di questa categoria di operatori: si sono evidenziati limiti derivanti dalla considerazione che l'arbitraggio, così come descritto dalla teoria, non trova spazio nella realtà dei mercati. L'arbitraggio difficilmente si presenta senza rischi a causa sia della mancanza di sostituti perfetti tra i titoli, sia della lentezza di convergenza dei prezzi ai valori insiti nelle informazioni.

Questa lentezza giustifica l'inerzia di cui soffrono i mercati nel reagire a comportamenti non razionali, correggendo gli squilibri.

Un terzo filone di critica deriva dagli studi della finanza comportamentale che, sebbene disciplina ancora in formazione, propone schemi di analisi complementari a quelli tradizionali.

Una nuova chiave di lettura potrà probabilmente arrivare dal metodo della simulazione che permetterà lo studio di questi fenomeni non in

maniera statistica a livello aggregato, ma partendo dall'ottica microeconomica. Questo in linea con il pensiero di tutti quegli studiosi che, come Samuelson, pensano che il mercato non sia efficiente a livello macro, ma lo sia a quello micro.

1.8 Teoria classica delle bolle speculative

Le convinzioni dei teorici dell'efficienza dei mercati sono messe a dura prova da dinamiche dei prezzi come quelle che hanno caratterizzato il biennio 1999-2000, al culmine di una lunga fase di crescita dei corsi.

Il normale andamento dei corsi della attività finanziarie alterna fasi di crescita ad altre di ribasso dei prezzi. Queste fasi sono spesso alternate da lunghi periodi di assenza di direzione delle quotazioni, con andamento laterale delle stesse. La continua crescita del benessere creato dall'uomo dovrebbe, a rigor di logica, portare ad un costante progresso dei corsi azionari, capaci di riflettere i miglioramenti economici del sistema. Una fase di crescita continua e decisa dei prezzi dei titoli azionari può essere considerata fisiologica in presenza di una contestuale crescita della reddività aziendale; allo stesso modo potrebbe essere considerata una lunga discesa delle quotazioni dovuta ad un peggioramento delle aspettative sul andamento della società.

Si può parlare di bolle speculative nei prezzi quando vi è un significativo allontanamento dei corsi azionari dai loro valori fondamentali, nonostante gli operatori possano ancora essere definiti razionali. Durante queste fasi di mercato, gli operatori percepiscono la divergenza tra prezzi stimati e prezzi correnti, tuttavia, si comportano come se ritenessero che la crescita dei titoli possa continuare ancora in futuro. Gli operatori abbandonano gli schemi di valutazione corretti basati sul valore attuale dei flussi futuri per adottarne altri che si basano sulla stima dei tassi di variazione attesi dei prezzi dei titoli. L'ottica temporale di questi investitori si ristringe, rendendoli sostanzialmente "miopi", incapaci di valutare correttamente il valore delle attività finanziarie.

Sin da questa definizione si intuisce come una bolla speculativa sia essenzialmente un fenomeno di natura psicologica che spesso abbandona gli aspetti macroeconomici del contesto di riferimento.

Come tutti i fenomeni anche le bolle nascono, si espandono e muoiono (*crash*). Molto spesso ciò avviene in maniera casuale con meccanismi che autoalimentano il rialzo dei corsi: la crescita dei prezzi conferma le aspettative, implicite nei rialzi precedenti, di ulteriori rialzi, allontanando sempre più il valore reale delle attività dalla loro quotazione.

Una simile dinamica non rimane costante, ma tende progressivamente ad accelerare, a causa dell'ingresso sul mercato di quanti sono attratti dalle prospettive di guadagni speculativi. A livello psicologico un simile comporta-

Figura 1.1: L'andamento mensile dell'indice COMIT Generale - Anni 1975-2002 (Fonte ADB Spa)

mento trova giustificazione dalla considerazione che la probabilità di ottenere rendimenti elevati dovrebbe compensare i nuovi entranti dalla possibilità di improvvisi crolli dei corsi. Questo comportamento riflette un modello ancora razionale: gli investitori accettano di essere sul mercato nonostante il rischio potenziale di una sopravvalutazione dei corsi e la grande volatilità che spesso ne caratterizza i movimenti.

L'ipotesi di fenomeni speculativi legati a bolle e *crash* nei prezzi azionari non è recente, ma si è formata sin dal momento in cui i mercati finanziari si sono organizzati. In letteratura sono descritti molti fenomeni che presentano situazioni di mercato in cui il prezzo sembra in contrasto con la spiegazione razionale. Il più famoso esempio di bolla speculativa è senza dubbio quello che interessò il mercato dei bulbi di tulipano nell'Olanda del XVII secolo e che viene citato come *tulipmania*. In quel periodo i coltivatori olandesi si stavano specializzando nella coltivazione e nella creazione di nuove varietà del fiore, importato pochi secoli prima dalla Turchia. Il mercato delle specie più rare registrò ben presto dei significativi incrementi dei prezzi, trasformando rapidamente i bulbi, da un lato, in forma di investimento, progressivamente ad appannaggio di investitori anche non professionali, e, dall'altro, in un oggetto in grado di conferire *status* ad alcune classi sociali. La convinzione che, anche in questo caso, i prezzi sarebbero continuati a crescere, spinse sempre più investitori ad entrare sul mercato, ormai dominato da schemi emotivi. Nonostante le ampie fluttuazioni dei corsi, influenzate dagli operatori professionali, avessero notevolmente incrementato il livello di rischio

del mercato, la fiducia degli operatori inizialmente impedì il ritorno verso il valore intrinseco dei prezzi.

Nel 1636, in pochi mesi, tuttavia, gli operatori più prudenti iniziarono a prendere consapevolezza di quanto fosse grande la componente di follia emotiva che aveva sorretto i corsi dei tulipani ed iniziarono a vendere. La crisi di fiducia che ne scaturì innescò un rapido crollo dei prezzi, alimentata da un'ondata di panico senza precedenti. Il crollo dei prezzi indebolì ulteriormente la fiducia degli investitori, con la conseguenza che alcuni acquirenti a termine, con contratti senza garanzie e stipulati prima del crollo, decisero di non rispettare gli obblighi. La gravità del fenomeno portò all'intervento delle autorità olandesi per riportare la calma sui mercati.

Accanto a questo episodio, la letteratura cita altri famosi fenomeni che vanno sotto i nomi di *Schema Ponzi*[6] e *catena di Sant'Antonio* che, in altri ambiti di applicabilità, riproducono, seppur con le dovute distinzioni, parte dei comportamenti alla base dei meccanismi di formazione delle bolle.

Più recentemente, numerosi studiosi hanno sottoposto a verifica l'ipotesi della presenza di bolle speculative nelle quotazioni azionarie del 1929, 1987 e 2000, verificando spesso elementi di similitudine con la dinamica del mercato dei tulipani dell'Olanda del XVII secolo.

Quanto descritto durante quegli anni sembra ricalcare, con le dovute sostituzioni, quanto avvenuto sui mercati azionari nei tre periodi citati e lo schema di creazione, sviluppo ed esplosione delle bolle sembra seguire sostanzialmente il modello originale.

La ipotesi di efficienza dei mercati ha cercato di coniugare le componenti irrazionali dei prezzi durante questi fenomeni, capaci di far allontanare i corsi dell'attività dal suo valore intrinseco, con l'ipotesi di razionalità degli operatori, che implica l'adozione dello schema delle aspettative razionali e l'ipotesi di rendimenti attesi costanti.

I modelli che sono stati sviluppati hanno preso vita dal paradigma delle aspettative razionali, in linea con le assunzioni alla base dei modelli di mercati efficienti. In linea con queste ipotesi, gli investitori, individui razionali, conoscono tutte le informazioni rilevanti per valutare il prezzo sulla base della funzione di probabilità dei rendimenti attesi. Gli investitori, quindi, non compiono errori sistematici nell'elaborazione delle previsioni. In questa cornice di riferimento, la possibilità che dalle azioni di questi operatori emerga e si sviluppi un processo assimilabile ad una bolla speculativa è legata alla ipotesi che la variazione attesa dei corsi diventi la variabile essenziale nella determinazione dei prezzi attuali. In questo caso, le aspettative sul prezzo del titolo abbandonano il legame con il valore intrinseco dell'attività,

[6]Il nome dello schema deriva da Charles Ponzi, spregiudicato gestore, che nel 1920 attrasse e poi truffò migliaia di risparmiatori che gli avevano affidato del danaro sulla base delle promessa di forti guadagni. Lo schema è ancora attuale: il danaro ricevuto non viene investito, ma serve a liquidare altri investitori inseriti all'interno di un pericolosissima struttura piramidale.

determinato dai suoi fondamentali, per assumere una correlazione positiva con il proprio tasso atteso di variazione futuro, in un processo iterativo che si autoalimenta.

Alcuni autori (si veda Caparrelli, 1986), partendo dalla determinazione del valore intrinseco della attività finanziaria, pari alla somma attualizzata dei flussi che origina nel periodo di possesso, sono giunti a modelli di rappresentazione delle bolle. Ipotesi di questi lavori è che gli investitori richiedano un rendimento atteso costante[7].

Nella valutazione del valore intrinseco del titolo per il periodo si parte dalla seguente formulazione:

$$P_t = \frac{E(D_{t+1} + P_{t+1})}{1+i}$$

in cui P_t e D_t sono rispettivamente il prezzo del titolo e il dividendo al tempo t, mentre i è il tasso di interesse costante. Questo modello rappresenta uno strumento di valutazione dei titoli coerente con l'ipotesi di *random walk* dei mercati finanziari e fornisce una formula di *pricing* se applicata iterativamente. In questo caso, con iterazioni successive si ottiene:

$$P_t^* = \sum_{s=1}^{\infty} \frac{E_t(D_{s+1})}{(1+i)^s}$$

dove P_t^* è il valore intrinseco del titolo, coerente con il noto modello del dividendo di Gordon.

L'equazione proposta avrà soluzione a condizione che:

$$\lim_{n\to\infty} \frac{E_t D_t + n}{(1+i)^t} = 0$$

che equivale alla ipotesi di convergenza dei prezzi al loro valore intrinseco.

Negando questa ipotesi, si ottengono un numero infinito di soluzioni pari a:

$$P_t = \sum_{s=1}^{\infty} \frac{E_t(D_{s+1})}{(1+i)^s} + B_t = P_t^* + B_t$$

definito in questo modo, il prezzo di mercato sarà pari alla somma del valore intrinseco P_t^* e di un ammontare B_t che sintetizza la componente di prezzo dovuta alla bolla speculativa. La bolla sarà pari alla deviazione del prezzo corrente dal suo valore reale.

Perché il modello abbia soluzione è necessario che la bolla corrente contenga l'aspettativa che questa è destinata ad estendersi nel tempo ossia che:

$$B_t = \frac{E_t(B_{t+1})}{1+i}$$

[7]Nel proseguo della trattazione indicato con $E_t R_t = k$ basato sull'assunto fondamentale che: $E_t(P_{t+1}) = P_t$.

Ipotizzando dividendi costanti e costanza della componente $B_t = b > 0$, la dinamica di crescita della bolla sarà interamente determinabile e sarà pari al tasso i. Sulla base delle precedenti formulazioni se ne deduce che dividendi e b, ipotizzati fissi, determinano la costante crescita del prezzo ad un tasso $(1 + i) > 1$. In forza della stabilità della componente legata ai dividendi, la componente del prezzo legata alla bolla assumerà progressivamente una proporzione crescente del prezzo corrente.

I modelli proposti dalla letteratura classica valutano la dinamica dei fenomeni speculativi legati a bolle e successivi *crash*, prescindendo da inizio e fine degli stessi. Le principali conclusioni a cui si può giungere indicano che:

a) la bolla non può essere negativa;

b) la bolla dopo essersi ridotta al valore nullo, può tornare a crescere;

c) si può escludere la possibilità di bolle per tutte quelle attività per cui esiste un limite maggiore di prezzo;

d) è sempre possibile che sui mercati azionari sia presente una bolla speculativa, ma l'ipotesi è plausibile solo quando l'orizzonte temporale del mercato è inferiore al tempo mancante all'esplosione della bolla.

Sulla base di questi schemi teorici, le bolle rappresentano un componente esogeno nel meccanismo di formazione del prezzo. Froot e Obstfeld (1991) hanno proposto un modello differente capace di valutare quelle che loro definiscono le "bolle endogene" le quali dipendono, in modo non lineare, dalle componenti del prezzo.

Il loro modello ipotizza l'elemento speculativo, rappresentato dalla bolla, dipendente dal flusso dei dividendi. Si ipotizza una relazione di correlazione positiva tra i due elementi. L'ipotesi appare coerente con l'evidenza dell'eccesso di reazione dei prezzi alle variazioni dei prezzi, di cui più volte è stato dato riscontro empirico.

Gli autori hanno distinto due diversi processi di formazione delle bolle endogene nei prezzi:

- il primo rappresenta un processo di tipo endogeno puro in cui P_t è definito come:

$$P_t = P(D_t) = P_t^* + B(D_t) = kD_t + cD_t^{\lambda}$$

dove D_t è il dividendo al tempo t mentre k, c e λ sono costanti. In questo caso i prezzi reagiscono più che proporzionalmente all'informazione sui dividendi, variando non solo in dipendenza della componente reale, ma anche di quella speculativa sulla base della relazione:

$$\frac{\delta P_t^*}{\delta D_t} = k + c\lambda D_t^{\lambda - 1} > k$$

ed essendo $c > 0$ la relazione conterrà una bolla che dipenderà esclusivamente dal valore dei dividenti e, pertanto, sarà da considerarsi endogena. La variazione del prezzo amplificherà il movimento del dividendo.

- il secondo processo è di tipo misto sulla base della relazione:

$$\tilde{P}_t = P_t^* + bD_t e^{(i-a-\frac{\sigma^2}{2})t}$$

dove i rappresenta il tasso di interesse istantaneo supposto costante, a e b sono costanti. In questo processo la componente speculativa è funzione dei dividendi e del tempo.

I risultati empirici raggiunti dai due studiosi hanno evidenziato la maggior capacità di allineamento al valore fondamentale P_t^* mostrata dalle bolle endogene che seguono il processo di tipo puro rispetto a quello di tipo misto; queste ultime sono state caratterizzate da un comportamento esplosivo nel tempo. Le verifiche compiute sulle bolle endogene di tipo puro hanno mostrato la presenza di un effetto di sovrareazione dei prezzi alla crescita dei dividendi e di un effetto di smorzamento della dinamica speculativa, con avvicinamento al prezzo corretto, in caso di diminuzione dei flussi dell'investimento.

Sebbene la teoria delle bolle speculative sembri abbastanza consolidata e capace di spiegare i fenomeni reali, i numerosi test di verifica compiuti empiricamente non sembrano aver dato risultati conclusivi.

Sono stati compiuti test sulla base dei modelli proposti da parte di numerosi studiosi senza ottenere una conferma definitiva ed integrale alle varie ipotesi.

Un ulteriore campo di ricerca su cui si sono confrontati gli studiosi è stato quello della verifica dell'eccesso di volatilità mostrato dalle quotazioni, cercando di valutare se le variazioni dei prezzi siano maggiori di quanto plausibile sulla base delle sole informazioni di carattere fondamentale. Se questa ipotesi fosse verificata, la volatilità in eccesso dei prezzi potrebbe essere frutto di movimenti irrazionali non dipendenti dalla componente fondamentale.

Nell'ambito di queste verifiche si è distinta particolarmente l'opera di Shiller (1981, 2000) che si basa su tre ipotesi fondamentali:

- le aspettative dei dividendi fondamentali influenzano il valore dei titoli;
- la costanza del rendimento reale atteso dell'investimento;
- i dividendi seguono un processo stazionario caratterizzato da un tasso costante di crescita.

Shiller ha inteso sottoporre a verifica l'ipotesi che se il prezzo di una attività finanziaria è uguale al valore attuale dei flussi di cassa che origina, allora è

possibile ipotizzare che il prezzo vari solo in risposta a cambiamenti delle aspettative sui flussi futuri.

Dal momento che i dividendi futuri sono difficilmente prevedibili, Shiller ha fatto ricorso all'ipotesi di aspettative razionali per ipotizzare la volatilità dei dividendi attesi pari a quella dei dividendi storici.

L'intento dell'autore è stato quello di cercare di individuare una soglia limite di volatilità del prezzo espresso dal mercato o della sua variazione. Lo studio è giunto alla conclusione che, dal momento che la varianza non può essere negativa, la volatilità del prezzo storico (indicato con P_t) deve essere minore o uguale a quelle del prezzo teorico ottenuto dall'attualizzazione dei flussi di dividendo futuri (P_t^*).

$$var(P_t) \leq var(P_t^*)$$

Secondo questa impostazione, la varianza dei corsi dei titoli è al massimo uguale a quella dei prezzi valutati secondo lo schema razionale *ex-post*.

$$var(P_t^*) \leq var(\sum_{s=1}^{\infty} \frac{D_{t+1}}{(1+i)^s})$$

Il risultato delle verifiche condotte, incapaci di confermare le ipotesi, ha mostrato come il mercato non sembri comportarsi in maniera razionale: i prezzi, secondo il test di volatilità proposto, non sembrano essere determinati unicamente dalla componente fondamentale della società emittente.

Il test di volatilità sembra negare l'ipotesi di efficienza dei mercati proposta da Fama, ma sulla fondatezza dei risultati ottenuti vi sarebbero forti perplessità di una parte della comunità scientifica. Il modello ha subito notevoli critiche in grado di mettere in discussione le conclusioni raggiunte.

1.8.1 Il modello proposto da Shiller

Nel tentativo di spiegare la bolla di fine millennio, Shiller (2000) ha proposto un modello di spiegazione del fenomeno che, facendo tesoro delle conclusioni raggiunte dalla teoria dei mercati finanziari, ne valuta anche componenti sociali e psicologiche in un'impostazione di finanza comportamentale[8].

Le analisi dello studioso sul comportamento dei mercati sembrano riprendere in parte il modello comportamentale collettivo proposto da Le Bon (1985), basato sull'osservazione della psicologia delle folle.

All'interno di questo schema teorico, l'autore individua il concetto di *feedback* come fondamentale per descrivere la nascita, lo sviluppo e l'esplosione di una bolla speculativa, con particolare attenzione ai fenomeni presenti sui mercati americani a cavallo del millennio.

I riferimenti empirici[9] ad altre due presunte bolle, quella del '29 e quella

[8] "... bolla speculativa: una situazione in cui i corsi temporaneamente alti sono in larga misura sostenuti dall'entusiasmo degli investitori e non da una stima coerente del valore reale." (Shiller, 2000)

[9] Particolare attenzione è dedicata all'analisi della dinamica del rapporto prezzo/utili.

degli anni sessanta, guidano l'autore all'individuazione di fattori acceleranti[10] che alimentano la crescita dei corsi e meccanismi amplificatori[11] che consentono all'euforia e alla dinamica della bolla di estendersi nel tempo.

Questi fattori agiscono, attraverso una fitta rete di retroazioni, su di un sistema caratterizzato da singoli investitori, affetti da illusioni cognitive, e da comportamenti gregari degli stessi che alimentano il contagio epidemico[12] delle condizioni alla base dello sviluppo della bolla speculativa.

L'impatto iniziale dei fattori acceleranti viene amplificato[13] in un aumento dei prezzi molto più ampio di quanto non fosse lecito attendersi sulla base delle condizioni iniziali. Una simile dinamica è spiegata dalla teoria della curva di retroazione (*feedback loop theory*) di cui esistono due versioni, entrambe parzialmente verificate a livello empirico:

a) la prima, quella che si basa sulle aspettative adattive, ipotizza che la retroazione abbia luogo perché aumenti passati dei prezzi generano aspettative di nuovi aumenti;

[10] Indicati da Shiller:

- la nascita di internet e più in generale di molteplici innovazioni tecnologiche;
- i fenomeni di "globalizzazione" e di riduzione della pressione concorrenziale sui mercati esteri;
- i cambiamenti nei valori culturali ora basati sulla ricerca del successo e del danaro;
- i cambiamenti nella legislazione fiscale;
- gli effetti del *baby boom*;
- l'attenzione crescente dei *mass media* per i mercati finanziari;
- le previsioni ottimistiche da parte degli analisti;
- l'impatto dei nuovi modelli pensionistici;
- la diminuzione dell'inflazione e gli effetti dell'illusione monetaria;
- i miglioramenti nell'efficacia e nell'efficienza degli scambi, dovuta alla diffusione dell'informatica nei sistemi di negoziazione dei titoli.

[11] Particolare attenzione viene posta su fattori in grado di alimentare le bolle, viste come schemi Ponzi naturali di tipo spontaneo:

- gli elevati livelli di fiducia (ottimismo) mostrati dai risparmiatori;
- aspettative ed emozioni degli operatori;
- l'attenzione del pubblico verso i mercati;
- la percezione della bolla e della retroazione negli investitori;
- molteplici fattori culturali.

[12] Esperimenti compiuti nel campo psicologico hanno dimostrato come le persone siano pronte a cedere al punto di vista maggioritario o all'autorità anche quando questi sono chiaramente in contraddizione con le valutazioni derivanti dai fatti. Nonostante tutto, questo comportamento è ritenuto razionale.

[13] Creando situazioni di euforia.

b) la seconda, quella basata sulla fiducia, ipotizza che quest'ultima aumenti in risposta ai precedenti rialzi dei corsi.

Indipendentemente dal meccanismo ipotizzato alla base della retroazione, la bolla non potrà continuare a gonfiarsi per sempre: non appena sarà cessata la domanda di titoli da parte degli investitori, la bolla sarà destinata a sgonfiarsi principalmente a causa della diminuzione dell'attenzione generale. Il processo di adeguamento e di correzione dei corsi potrebbe essere lento in forza della ipotesi che lo scoppio delle bolle non debba essere improvviso.

Lo schema teorico delineato prevede anche la possibilità che si formino bolle negative, caratterizzate da una retroazione capace di spingere i titoli al ribasso, in conseguenza a precedenti discese dei corsi.

Il processo di crescita (ribasso) dei corsi continuerà finché ulteriori rialzi (diminuzioni) cominceranno a diventare improbabili.

I fenomeni generati dalla curva di retroazione possono sviluppare comportamenti complessi, e persino potenzialmente casuali, in virtù di relazioni non lineari che coinvolgono le variabili.

Shiller cita, come modello di retroazione applicabile alla dinamica e al comportamento delle bolle, gli schemi Ponzi naturali, di tipo spontaneo, che nascono dalla trasmissione dell'illusione tra gli operatori.

Attori di questa struttura sono operatori che, sulla base di conclusioni ottenute da ricerche psicologiche, agiscono in linea con comportamenti non del tutto razionali, espressione allo stesso tempo di forza e limiti delle capacità intellettive degli esseri umani. Alla base di queste azioni, Shiller individua "ancore morali" e "ancore quantitative" che inducono gli operatori a compiere determinate decisioni, subendone gli effetti.

La conclusione a cui giunge l'autore è che l'ipotesi del mercato efficiente non è forte abbastanza da garantire al mercato di non poter essere caratterizzato da periodi di errata valutazione delle attività finanziarie; queste fasi si sono mostrate capaci di estendersi per periodi più o meno lunghi. Il comportamento più o meno razionale degli operatori può non essere sufficiente a riportare in equilibrio i prezzi delle attività, correggendo gli scostamenti.

Queste conclusioni confermano i limiti, individuati da Shiller, all'ipotesi di efficienza dei mercati. Secondo l'economista:

> In conclusione, i prezzi delle azioni vivono una vita propria: non rispondono *tout court* agli utili o ai dividendi. E non sembra neppure che essi siano determinati esclusivamente dalle informazioni circa i futuri utili o dividendi. Per trovare spiegazioni sui movimenti dei corsi , dobbiamo rivolgerci altrove. (Shiller, 2000)

Figura 1.2: L'andamento mensile dell'indice NASDAQ Composite - Anni 1982-2002 (Fonte ADB Spa)

1.9 Conclusioni

Gli argomenti contrari all'idea che il comportamento dei mercati possa essere affetto da bolle e *crash* si fondano sulla base intellettuale costituita dalla teoria di efficienza dei mercati e delle numerose ricerche a sostegno di questa tesi. Secondo l'ipotesi di efficienza dei mercati nelle sue versioni, i prezzi dei titoli incorporano, in ogni momento, set di informazioni più o meno ampi. I titoli sono sempre correttamente valutati sulla base dei prezzi così determinati. Le quotazioni di titoli che a volte possono sembrare eccessivamente "alte" o "basse", in realtà, adottando il punto di vista della teoria sarebbero delle semplici illusioni. Nel tempo, i prezzi dei titoli seguono percorsi casuali (*random walk*); a determinare i movimenti dei corsi sono le informazioni che via, via vanno ad arricchire il set informativo, ed essendo nuove, sono imprevedibili per definizione.

La teoria così definita è stata sottoposta a numerosi test usando i dati forniti dai mercati finanziari, e nonostante sia stata confutata da alcuni studiosi, continua sostanzialmente ad essere ritenuta veritiera.

Accanto a questi strumenti classici di analisi, i ricercatori stanno cercando nuovi metodi di comprensione e rappresentazione della realtà dei mercati finanziari. Il metodo della simulazione, proposto nei capitoli seguenti, si presenta potenzialmente in grado di dare nuove risposte alla teoria dei mercati finanziari nella loro dimensione di sistemi complessi. I risultati sperimentali descritti nell'ultimo capitolo sembrano dare una prima, seppur parziale,

conferma a queste attese.

Capitolo 2

Simulazione e modelli ad agenti

> "... La scienza non è una raccolta di leggi, un catalogo di fatti senza nesso. E' una creazione dell'intelletto umano, con le sue libere invenzioni d'idee e di concetti. Le teorie fisiche tentano di costruire una rappresentazione della realtà e di determinarne i legami con il vasto mondo delle impressioni sensibili. Pertanto le nostre costruzioni mentali si giustificano soltanto se le teorie costituiscono realmente un legame di tal fatta e secondo come lo costituiscono. (...)
>
> Con l'aiuto delle teorie fisiche cerchiamo di aprirci un varco attraverso il groviglio dei fatti osservati, di ordinare e di intendere il mondo delle nostre impressioni sensibili. Aneliamo a che i fatti osservati discendano logicamente dalla nostra concezione della realtà. Senza la convinzione che con le nostre costruzioni teoriche è possibile raggiungere la realtà, senza convinzione nell'intima armonia del nostro mondo, non potrebbe esserci scienza. Questa convinzione è, e sempre sarà, il motivo essenziale della ricerca scientifica. In tutti i nostri sforzi, in ogni drammatico contrasto tra vecchie e nuove interpretazioni riconosciamo l'eterno anelo d'intendere, nonché l'irremovibile convinzione nell'armonia del nostro mondo, convinzione ognor più rafforzata dai crescenti ostacoli che si frappongono alla comprensione." (Einstein-Infeld 1938)

Frutto delle teorie epistemologiche elaborate dal Seicento in poi, il metodo di analisi della scienza moderna trova la propria definizione grazie alla rivoluzione avviata dall'opera di Galileo e Newton. La rivoluzione del pensiero scientifico che culminò con Newton[1] condusse a vedere nell'universo

[1] Il cui pensiero raccolto nella famosissima opera *Philosophiae naturalis principia*

una sorta di gigantesco meccanismo "preciso come un orologio". Secondo questa concezione il comportamento del meccanismo poteva essere perfettamente predetto sulla base dell'assunto che in condizioni identiche avrebbe fatto sempre cose identiche.

Identificato il compito dello scienziato come quello di conoscere e capire la realtà[2], allora la scienza può essere definita (Parisi, 2001) come "l'interfaccia tra teorie e osservazioni empiriche". Fine ultimo della scienza è quello di costruire un quadro unico o unificato della realtà in cui ogni fenomeno sia collegato in maniera diretta o indiretta con tutti gli altri, senza alcuna discontinuità sulla base delle regolarità del loro comportamento.

Nella scienza sono le predizioni assoggettabili alla verifica tramite i fatti direttamente osservati (*empirici*) ad unire realtà e teorie. Grazie al momento della verifica si caratterizzano e si distinguono i processi cognitivi di tipo filosofico da quelli propriamente scientifici; questi devono soddisfare alcuni requisiti per rendere la scienza il più possibile oggettiva. Il processo deve essere replicabile, dando ad un qualunque studioso la possibilità di ottenere gli stessi risultati dalle stesse osservazioni e dagli stessi esperimenti (metodo degli esperimenti di laboratorio). Inoltre, il processo deve portare alla definizione di teorie formalizzate matematicamente in grado di cogliere in maniera precisa e puntuale gli aspetti quantitativi dei fenomeni spiegati.

Se questo paradigma ha trovato piena applicabilità in fisica, chimica e biologia, le cosiddette *"scienze della natura"*, non altrettanto è avvenuto per quelle *"dell'uomo"* in cui quasi mai si è avuta l'interazione tra fatti empirici e teorie. Private molto spesso di formulazioni matematiche da sottoporre al metodo degli esperimenti, queste scienze hanno registrato minori progressi rispetto a quelle della natura. Le necessità di introdurre semplificazioni (talvolta poco plausibili) di realtà troppo complesse per essere ridotte in termini matematici hanno limitato le capacità di sviluppo di teorie da parte degli studiosi. A dare nuove possibilità di riscatto a queste scienze arriva il metodo simulativo che grazie all'uso delle tecniche di elaborazione elettronica apre nuove possibilità di sviluppo finora inesplorate.

Ma una nuova rivoluzione del paradigma scientifico sembra profilarsi all'orizzonte. I "crescenti ostacoli che si frappongono alla comprensione citati da Einstein erano rappresentati dalle crescenti difficoltà della scienza nello spiegare alcuni fenomeni della fisica delle particelle. Le inquietudini dello scienziato austriaco erano il frutto delle teorie probabilistiche introdotte dalla meccanica quantistica per definire velocità e posizione delle singole particelle, sulla base del principio dell'indeterminazione. Quello che fino ad allora era stato un orologio perfetto non poteva più essere considerato tale in seguito alla scoperta che sistemi in grado di obbedire a leggi immutabili

matematica può essere sintetizzato dal messaggio: "la natura ha le sue leggi, e noi possiamo trovarle."

[2]Per poter giungere a prevederne e controllarne gli schemi di funzionamento.

e precise non sempre agiscono in modi precisi e regolari, producendo comportamenti che appaiono casuali e l'emergere di fenomeni non previsti o imprevedibili a priori[3].

Questa scoperta straordinaria modifica sensibilmente il pensiero scientifico, determinando pesanti implicazioni sulle nozioni citate in precedenza di predizione e replicabilità dei risultati, con l'effetto di ribaltare la dicotomia semplice-complesso. Si scopre così che fenomeni che possono sembrare casuali e privi di struttura possono obbedire in realtà a leggi semplici, dando vita ad un caos[4] deterministico molto spesso identificato con il termine *complessità*[5].

2.1 Che cosa è una simulazione

La simulazione, come detto in precedenza, rappresenta uno strumento di esplorazione del reale, profondamente diverso da quelli tradizionali ed in grado di determinare profonde conseguenze particolarmente nella conoscenza e nella comprensione dei fenomeni umani.

In primis, la simulazione costituisce un nuovo modo di esprimere le teorie scientifiche, non più limitate ad essere formalizzate con concetti ed idee astratte e comunicate in termini di simboli verbali. Le difficoltà e le restrizioni della comunicazione attraverso un linguaggio fatto di simboli hanno spesso

[3] "Lorenz si accorse che le sue equazioni non si comportavano nel modo in cui si sarebbe atteso un matematico tradizionale, e coniò dunque la sue famosa espressione "effetto farfalla". Il battito delle ali di una sola farfalla produce un minuscolo mutamento nello stato dell'atmosfera. In capo ad un certo periodo di tempo, il comportamento dell'atmosfera diverge da quello che sarebbe stato senza quel battito d'ali. Ne consegue che, a un mese di distanza, non si verifica un tornado che avrebbe devastato le coste dell'Indonesia. Oppure, viceversa, si verifica un tornado che non ci sarebbe stato.

Ci sono farfalle dappertutto. Ma chi può dire che gli effetti dei battiti delle loro ali si cancellino reciprocamente?" (Stewart, 1993)

[4] Una delle definizioni presenti riportate dal *Dizionario enciclopedico Italiano dell'Istituto Treccani*, identifica il caos come un "Comportamento stocastico che si verifica in un sistema deterministico", in altri termini può essere definito come un comportamento senza legge governato per intero dalla legge.

[5] Secondo la definizione data dai ricercatori del Santa Fe Institute la complessità è caratterizzata da:

- interazioni diffuse tra una moltitudine di agenti eterogeni che agiscono localmente in un determinato contesto;
- assenza di meccanismi di controllo e coordinamento sulle azioni dei soggetti e sul comportamento complessivo del sistema;
- emergenza di una organizzazione gerarchica trasversale tra gli elementi;
- capacità di adattamento continuo degli agenti e del sistema nel suo complesso;
- capacità di creazione continua di nuovi comportamenti e strutture;
- assenza di equilibrio.

.

costituito un vincolo al progresso scientifico, costretto a tener conto della semantica delle singole espressioni e vincolato alla semplice "spiegazione" del reale. Queste difficoltà sono solo in parte state risolte dall'utilizzo di simboli quantitativi della matematica e di espressioni o schemi grafici in grado di dare rappresentazioni più oggettive alle teorie descritte rispetto al semplice uso delle parole.

La simulazione supera questi limiti perché nel formalizzare una teoria non usa simboli destinati ad essere interpretati da altri scienziati: la stessa teoria è espressa come programma di computer che ne incorpora concetti, processi e postulati. L'esecuzione del programma grazie all'elaboratore elettronico, interpretando i simboli informatici[6] scritti nel codice sorgente, riprodurrà la realtà simulata che non sarà solo più spiegata e predetta ma addirittura ricreata.

Spezzando il collegamento immediato tra un simbolo e la sua semantica o meglio filtrandolo attraverso l'uso del codice dell'elaboratore elettronico viene ad essere rivoluzionato il rapporto concatenato rappresentabile dal triangolo costituito da teorie, predizioni empiriche e mente dello scienziato. I simboli dei linguaggi formali con cui sono scritti i programmi dei computer non rappresentano simboli nello stesso senso con cui vengono definiti quelli dotati di semantica per la mente umana e utilizzati nelle teorie scientifiche classiche.

Queste ultime rappresentano schemi mentali formalizzati degli scienziati nello studio di un fenomeno e costretti ad essere riprodotti, in maniera "passiva", da altre menti per essere interpretati e capiti nel loro significato. Per contro, le simulazioni rappresentano teorie "attive" o per meglio dire "vive" capaci di ricreare fenomeni simulati corrispondenti con quelli reali.

Ruolo della scienziato-simulatore sarà proprio quello di verificare questa corrispondenza, osservando fenomeni reali e simulati.

Il processo di verifica della corrispondenza, e delle relative ipotesi, avverrà osservando i fenomeni riprodotti in condizioni controllate, ed eventualmente manipolate, all'interno della simulazione che viene a costituire una sorta di ''laboratorio sperimentale virtuale" (Parisi 2001). All'interno del laboratorio così costruito, significamene più efficiente e flessibile di quelli reali, potrà avvenire quel fondamentale confronto tra le predizioni empiriche con la realtà che permette la verifica sperimentale della teoria. Ciò è possibile perché le predizioni empiriche derivate dalla teoria sono gli stessi risultati della simulazione, i fenomeni simulati che vengono prodotti dall'elaboratore elettronico. In questo modo, la teoria risulta incorporata nella simulazione attraverso il linguaggio simbolico-formale che costruisce il codice del programma.

Se questo listato di istruzioni rappresenta la stessa teoria e quindi il punto di arrivo del processo cognitivo, la simulazione è un valido aiuto an-

[6]Questi simboli vengono definiti formali o "sintattici" (cioè non semantici).

che nella fase di elaborazione della teoria, sviluppando e valorizzando nuove metodologie di ricerca come nel caso del metodo degli *esperimenti mentali*[7]. Il processo cognitivo di uno scienziato-simulatore è profondamente diverso da quello di uno scienziato che utilizza gli schemi tradizionali: per il primo il processo di ricerca implica l'elaborazione contemporanea e congiunta di teoria e simulazione, con un incessante e continuo scambio di idee e ipotesi tra le due fasi. La teoria espressa a livello di *esperimento mentale* vede le proprie predizioni immediatamente verificate o negate dai risultati espressi dalla simulazione, con una nuova forma di conferma empirica. E in virtù della corrispondenza tra i risultati e i fenomeni reali, l'esperimento mentale sarà o meno confermato. Questo caratterizza un tratto distintivo dell'utilizzo del metodo simulativo che ha profondi impatti sulla natura delle teorie elaborate e sulla realtà artificiale riprodotta.

Se le scienze della natura, la cui evoluzione recente è stata strettamente connessa con l'introduzione e lo sviluppo dell'informatica, hanno saputo adottare la simulazione come strumento di studio, non altrettanto hanno saputo fare le scienze dell'uomo, per cui la nuova metodologia di analisi risulta ancora piuttosto nuova.

Fra gli studiosi di queste ultime, recentemente, è cresciuta l'attenzione dedicata alle possibilità aperte dalla simulazione nella ricerca di nuove teorie. Ciò è potuto avvenire come sottolinea Conte (1997):

- per le sempre maggiori difficoltà incontrate dagli strumenti di ricerca tradizionali nel comprendere la dinamica sociale e nel formalizzare e verificare le proprie teorie;
- per l'importanza sempre maggiore attribuita alle strutture sociali e all'emergere di modelli comportamentali;
- per il crescere dell'attenzione attribuita ai fenomeni sociali di autorganizzazione.

Il combinarsi di questi elementi ha favorito l'estendersi dell'applicazione della metodologia della simulazione anche nelle scienze dell'uomo, dotandole di un autentico metodo "scientifico".

[7] "Un esperimento mentale è un esperimento non condotto realmente, osservando e manipolando la realtà, me è un esperimento immaginato. Sia in un esperimento reale, che in un esperimento mentale, lo scienziato mette alla prova le sue idee e le sue teorie sulla realtà, e più precisamente le idee e le predizioni tratte da queste idee e teorie. Ma mentre nell'esperimento reale lo scienziato si basa sulla osservazione della realtà, sulla manipolazione fisica di cose fisiche, e sulle conseguenze osservate di queste manipolazioni, in un esperimento mentale tradizionale lo scienziato si limita a immaginare delle cose, delle manipolazioni di tali cose, e i risultati, anch'essi immaginati, di queste manipolazioni immaginate." (Parisi, 2001)

2.2 I vantaggi della simulazione

L'adozione della simulazione come strumento di analisi comporta per gli studiosi una serie di vantaggi, dettati dalle peculiarità del metodo stesso capace, come evidenziato nel capitolo precedente, di costituire una sorta di sintesi tra i metodi tradizionali dell'*induzione* e della *deduzione*.

L'utilizzo della simulazione, formalizzata in un programma per l'elaboratore, per derivare le stesse predizioni empiriche delle teorie ne rende oggettivo il momento della verifica, sia esterna che interna. Ciò è particolarmente vero per quest'ultima tipologia di verifica a cui deve essere sottoposta una teoria scientifica e con la quale si tende a stabilire se una predizione empirica deriva espressamente dalla teoria proposta. L'importanza di questa fase discende dal fatto che solo in caso di esito positivo della verifica interna, i fatti che verificano la predizione possono essere effettivamente spiegati dalla teoria. L'uso della simulazione rende automaticamente oggettiva questa fase, liberando la mente del ricercatore da ogni dubbio sulla capacità della propria simulazione di rappresentare la realtà.

Inoltre, l'opera dello studioso che ricorre al nuovo strumento è decisamente facilitata non solo in fase di *test*, ma durante l'intero processo di analisi del fenomeno. La necessità di utilizzare l'elaboratore elettronico per derivare le predizioni empiriche dalle teorie elimina gli errori durante questo processo, rendendolo oggettivo e riproducibile come un laboratorio sperimentale artificiale. Il vantaggio è ancor più significativo per le discipline sociali per cui sono evidenti i benefici rispetto all'uso di teorie espresse esclusivamente in termini quantitativi o qualitativi.

Proprio l'uso della simulazione come un laboratorio sperimentale artificiale comporta notevoli vantaggi al processo cognitivo per tutte le scienze e in particolar modo quelle sociali, impossibilitate a studiare i propri fenomeni utilizzando laboratori reali. La simulazione, inoltre, rappresenta la possibilità di studiare fenomeni complessi senza dover ricorrere a suddivisioni astratte di parti o fattori isolati dal tutto. Liberato da queste semplificazioni artificiali, lo studioso del fenomeno è in grado con il proprio modello simulato di spiegarne il funzionamento complessivo, cogliendo la reale portata delle interrelazioni e delle interconnessioni tra i fattori, e non soltanto il comportamento di singoli aspetti.

Con la simulazione, lo studio della realtà nel suo insieme non determina limiti o perdite di efficienza in termini di controllo e di manipolabilità dei fattori, che nel laboratorio sperimentale artificiale possono essere opportunamente gestiti dallo scienziato senza alcun vincolo.

Inoltre grazie alla simulazione non solo è possibile ricreare il reale presente e passato, ma è anche possibile costruire dei mondi che non esistono nella realtà, ma sono solo "in potenza". Si creano dei mondi possibili o probabili che nascono dalla mente dello studioso e dalla sue intuizioni su di

una teoria esplicita che può essere sottoposta a verifica esclusivamente con il metodo sperimentale.

Ma è soprattutto nello studio dei *sistemi complessi* che le simulazioni forniscono un apporto significativo al processo cognitivo. La mente umana sviluppa il proprio pensiero sui fenomeni grazie alla capacità di astrarre[8] dalla realtà il nesso di causalità tra gli eventi. La capacità di astrazione fa tendere la mente a concepire il reale come una somma di *sistemi semplici*[9], in cui una singola causa implica un singolo effetto sulla base di una relazione "lineare". La linearità è ancor più evidente quando a determinare un effetto intervengono più cause: in questo caso si tende separare il ruolo svolto da ciascun fattore che risulta isolato e prevedibile senza valutarne gli effetti in termini di interazione. Sistemi semplici definiti in questo modo risultano facilmente prevedibili nella loro evoluzione dal momento per conoscerne lo stato in un determinato istante sarà sufficiente conoscerne le condizioni al momento precedente. Anche l'effetto di eventuali perturbazioni esogene sarà facilmente predicibile e direttamente commisurato all'entità dell'evento esterno. In questo modo, l'evoluzione e lo sviluppo di un sistema semplice dipende profondamente dalle sue condizioni iniziali e queste dinamiche possono essere isolate dal contesto in cui si trova. Tra le condizioni iniziali e gli altri fattori di un simile sistema non ci saranno rapporti di causazione reciproca, ma i nessi di causa-effetto saranno unidirezionali senza alcuna retroazione. Ogni elemento del sistema svolge una ruolo ben definito all'interno del funzionamento complessivo del tutto e l'effetto di ogni elemento è facilmente individuabile.

Risulta chiaro che utilizzando questa ottica di rappresentazione della realtà, ogni sistema semplice può essere inserito in una gerarchia di sistemi, alcuni dei quali potenzialmente uguali tra loro, che partendo da un livello di osservazione particolare della realtà allarga il campo di studio a quello generale.

Nonostante gli indubbi vantaggi in termini di semplificazione e di analisi[10] che un simile modo di rappresentazione comporta, adattandosi perfettamente al modello cognitivo degli esseri umani, la sua scarsa capacità di corrispondenza con il reale ne evidenzia pericolosamente i limiti.

Nel reale, i sistemi semplici rappresentano l'eccezione in un mondo dominato da relazioni tra cause ed effetti non lineari. Il concorso di più cause nel produrre un dato effetto e i meccanismi di retroazione e di interconnes-

[8] Sacrificando spesso l'eterogeneità e le diversità dei fenomeni reali oggetto di studio.

[9] Un sistema semplice è lineare, il suo comportamento complessivo dipende dalla somma delle azioni delle sue componenti secondo un principio di sovrapposizione; per contro in un sistema complesso tale principio non agisce e il comportamento del tutto non coincide con la semplice somma di quello delle parti.

[10] "La scienza tradizionale è analisi della realtà, dove analisi significa partire dalla realtà, scomporla nelle sue componenti e poi ricostruire i fenomeni della realtà mettendo insieme queste componenti con la teoria e con il ragionamento."(Parisi 2001)

sione tra gli effetti rendono vano il tentativo di considerare indipendente ogni evento e di isolarne le implicazioni. Quando descritto è tipico dei *sistemi complessi*, costituiti da moltissimi elementi, spesso diversi tra loro, ma perfettamente conosciuti allo studioso, che si influenzano reciprocamente a livello locale con interazioni anch'esse note. Proprio da queste interazioni tra gli elementi di portata limitata, ex-post, emergono le proprietà e i comportamenti globali dell'intero sistema che non sono deducibili o prevedibili a priori dall'osservatore.

Risulta evidente il carattere antitetico tra sistemi complessi e sistemi semplici, che differiscono anche per altre importanti proprietà; se da un lato, queste mettono in risalto le innegabili difficoltà per gli studiosi nell'analisi di sistemi non lineari, dall'altro, enfatizzano la manifesta superiorità dei sistemi complessi nella capacità di aderire alla realtà oggetto di esplorazione. Quanto affermato in precedenza è soprattutto vero per le scienze sociali in cui i fenomeni molto spesso si sviluppano in sistemi complessi.

Lo stato futuro di un sistema complesso, come accennato in precedenza, non può essere facilmente prevedibile sulla base della conoscenza delle sue condizioni passate: in un sistema caotico la notevole sensibilità alle condizioni iniziali e gli effetti di variazioni esogene dei fattori non commisurati all'entità della perturbazione rendono ardua una valutazione ex-ante sulla evoluzione del sistema. Ciò fa perdere controllabilità e comprensibilità al modello, alimentando le incertezze dello scienziato che, nel suo operato, si trova privato delle importanti garanzie fornite dalla ipotesi di replicabilità del sistema e dei risultati ottenuti.

Inoltre, un sistema così definito, dipendendo dal contesto in cui opera, è di tipo aperto: gli scambi con l'ambiente, spesso costituto da gerarchie di sistemi interdipendenti, influenzano un mondo in cui i singoli elementi sono interconnessi da rapporti di causazione reciproca, rendendo vano il tentativo di identificazione della funzione svolta da ciascuna parte nel comportamento del tutto.

Gli strumenti tradizionali della scienza, sviluppatisi per l'analisi del semplice e del lineare, mostrano tutti i loro limiti nello studio dei sistemi complessi: le limitazioni dei simboli del linguaggio nell'esprimere concetti e teorie complesse sono il riflesso dei limiti nel processo cognitivo degli esseri umani, incapaci di rappresentare simbolicamente modelli complicati in cui una molteplicità di parti interagisce, influenzandosi reciprocamente[11]. La limitazione è evidente se si pensa agli strumenti messi a disposizione dal linguaggio dei numeri: gran parte della teoria matematica si fonda sull'ipotesi di linearità funzionale, di equilibrio statico e sulle possibilità di ottimizzazione dei fattori.

Non solo il linguaggio della scienza mostra i propri limiti, ma lo studio di sistemi complessi rende inutilizzabile anche il metodo degli esperimenti

[11] Perdendo l'attenzione sull'aspetto dinamico dei sistemi.

di laboratorio in senso classico, fondato su replicabilità e controllabilità di condizioni e risultati.

Privato di gran parte degli strumenti di analisi tradizionali e costretto a confrontarsi con una realtà in gran parte formata da sistemi complessi, lo scienziato ha dovuto ricercare un nuovo apparato metodologico, quello delle simulazioni, che, grazie all'uso dell'elaboratore elettronico capace di sopperire ai limiti della mente umana, rende agevole la sintesi[12] della realtà di questi fenomeni. Teorie complesse che, come detto in precedenza, saranno incorporate nel codice del programma della simulazione, verificata positivamente solo se capace di riprodurre il reale.

L'aver adottato logica dei sistemi semplici ha portato, come detto in precedenza, all'analisi della realtà non in maniera unitaria ma profondamente frammentata: ciò ha avuto un notevole riflesso nella frammentazione dei metodi e delle discipline che caratterizza la scienza. Grazie all'adozione della simulazione, metodo proposto per lo studio di fenomeni di qualunque tipo, la scienza ha ritrovato la propria capacità di sintesi, superando la logica della specializzazione delle discipline per adottarne una nuova interamente nuova e multidisciplinare. La simulazione, in questo modo, può essere vista come un linguaggio comune tra gli scienziati, in grado di riportare unità tra le discipline scientifiche.

2.3 Critiche e limiti della simulazione

Come ogni metodo di ricerca anche la simulazione presenta limiti e svantaggi che sono stati posti in luce dalla comunità scientifica, spesso rigida di fronte a possibili rivoluzioni eterodosse del metodo. Nonostante gli indubbi vantaggi che l'adozione del metodo di ricerca della simulazione è in grado di apportare nella comprensione del reale, la sua diffusione ha risentito delle critiche sui limiti sollevati dagli studiosi. Queste critiche sono spesso state mosse più per il frutto dalla mancata comprensione delle potenzialità e delle assunzioni teoriche del metodo che per i reali limiti dello stesso.

Per citare le più frequenti obiezioni infondate che vengono sollevate alla simulazione si possono riportare le accuse di:

- semplificare eccessivamente la realtà;
- non dire nulla di nuovo, limitandosi a riprodurre i fenomeni;
- di simulare una realtà che non si conosce ancora bene;
- di comportarsi come una "scatola nera" che riproduce un fenomeno senza spiegarlo in maniera esplicita.

[12] "Sintesi vuol dire partire dalle singole componenti per studiare cose emerge quando queste componenti vengono messe insieme e fatte interagire."(Parisi 2001)

Dalla lettura del paragrafo precedente in cui il metodo simulativo è stato definito risulta evidente la pochezza di simili argomentazioni, derivanti dalla scarsa conoscenza su mezzi e scopi del paradigma proposto.

Ben più importante è in questa sede evidenziare le limitazioni di cui soffre il metodo e che sono generalmente riconosciute dagli stessi utilizzatori.

Un primo problema delle simulazioni è lo scarso peso attribuito al ruolo della verifica esterna delle teorie, spesso subordinata alla ricerca di forme di coerenza interna tra formulazioni teoriche e predizioni empiriche che da essa dovrebbero derivare[13]. Molto spesso superato il momento della verifica interna, lo scienziato-simulatore non verifica la effettiva corrispondenza tra i risultati della simulazione e la realtà empirica, chiudendo il circolo del processo cognitivo.

Portando alle estreme conseguenze questa limitazione, il rischio diventa quello prospettato da Pryor (2000) in un lavoro auto-ironico, in cui un autore sconosciuto nel 2028, dopo la caduta di un asteroide sulla terra, ritrova un libro sulla complessità degli anni '90. Dopo una attenta lettura, l'autore catalogherà il ritrovato come "un tipico libro" sull'argomento di anni in cui "quasi tutti i lavori non contengono applicazioni empiriche reali, a parte qualche interessante aneddoto".

Inoltre, a minare la effettiva corrispondenza tra i risultati della simulazione e la realtà empirica interviene un uso non sempre accorto delle semplificazioni che vengono introdotte. Ad essere in discussione non è tanto l'uso dello strumento della semplificazione per comprendere e rappresentare meglio la realtà, quanto la scarsa attenzione dedicata alla distinzione tra aspetti rilevanti, da includere nel modello, e fattori irrilevanti ai fini della spiegazione del fenomeno simulato.

Non sono solo le semplificazioni introdotte per sopperire alle lacune cognitive dello studioso a costituire un fattore critico in grado di verificare o meno i risultati della sua opera. Altrettanto critici risultano i dettagli arbitrari che debbono essere inseriti nella simulazione per poter costituire un sistema funzionante all'interno dell'elaboratore elettronico. In questo caso, la tecnologia svolge un ruolo determinante nel guidare le scelte dello studioso che dovrà comunque tener ben presenti gli scopi pratici e quelli della conoscenza dei propri studi.

2.4 Le scienze dell'uomo e la simulazione

Valutati appieno limiti e prospettive del metodo della simulazione, risultano evidenti le grandi potenzialità che un simile strumento abbia per il sapere scientifico nel proprio complesso ed in particolar modo per le discipline "dell'uomo".

[13] Ossia la verifica interna della capacità della teoria di spiegare effettivamente certi fatti empirici.

A determinare il progredire difficoltoso di queste ultime interviene non solo l'assenza di un profondo interscambio tra teoria ed osservazione empirica, ma anche una serie di cause imputabili principalmente alle peculiarità del loro oggetto di studio. Studiare i fenomeni umani presenta elementi di grande diversità rispetto allo studio di quelli naturali: ad essere osservati sono gli esseri umani ed i loro comportamenti, frutto di una "vita mentale" che risponde a schemi soggettivi e guidata dalla libertà di azione. Proprio questi aspetti rendono difficile sviluppare una scienza di tali fenomeni che sia paragonabile alle scienze di natura.

Ciò è ancor più vero se si valuta l'incapacità di tali scienze di far interagire nel modo appropriato teorie con i dati empirici attraverso un dialogo costante, tipico dello studio dei fenomeni naturali. La capacità delle varie scienze di creare questo dialogo non è stata univoca e a questo riguardo occorre introdurre una opportuna classificazione dei risultati raggiunti dalle varie discipline[14]:

1. alcune scienze come la storia e l'antropologia culturale consentono osservazioni empiriche, ma non la formalizzazione di teorie;

2. altre come la sociologia sono dotate sia di teorie sia di osservazioni empiriche, ma senza alcuna interazione tra queste;

3. ed infine, alcune come l'economia e la linguistica formale che hanno saputo creare un dialogo, più o meno intenso, tra teoria ed osservazione empirica, nonostante i vincoli imposti dalle particolarità dei fatti empirici osservati spesso in numero ristretto abbiano favorito l'adozione di apparati teorici talvolta statici e superficiali. Per quanto riguarda l'economia, recentemente lo sviluppo dell'econometria ha dato un significativo impulso all'applicazione della statistica nell'analisi dei fenomeni di questa disciplina.

Proprio l'economia[15] ci fornisce un utile esempio dei limiti incontrati dal progredire delle scienze dell'uomo. Gli economisti per descrivere comportamenti aggregati a livello sistemico hanno dovuto ricorrere a complesse espressioni quantitative per rappresentare le relazioni che si osservano empiricamente tra le singole variabili. Andando oltre il livello aggregato sono state introdotte artificiose quanto irrealistiche semplificazioni sul comportamento razionale dell'essere umano. Ecco allora spiegata l'introduzione dell'agente rappresentativo, dotato di perfetta razionalità in grado di ottimizzare[16], con

[14] Le scienze sociali sono ripartite in una artificiosa quanto pericolosa "suddivisione in discipline che non corrisponde ad alcun processo naturale e appare largamente arbitraria." (Parisi, 2001)

[15] Definita da Castelfranchi (1996) "una scienza truccata. Fa finta di essere scienza generale e astratta delle decisioni razionali."

[16] "La trappola dell'ottimizzazione" come definita in Terna (1996), riportando il pensiero di Shoemaker (1991): "Nelle mani di un abile economista matematico, un numero

o senza vincoli, i comportamenti sulla base della propria funzione di utilità (Harley, 1996). Tale concezione, corretta grazie all'opera di Simon (1969) con cui si è introdotto il principio della razionalità limitata, è ancora oggi alla base di molte teorie economiche.

Con la mera illusione di spiegare i fenomeni osservati, il ricorso a questi strumenti ha portato gli studiosi a costruire modelli astratti, lontani da una realtà che "probabilmente è simultaneamente: più semplice, nei soggetti; più complessa nelle iterazioni." (Terna, 1996) Il riferimento al metodo della simulazione per coniugare questi aspetti e superare i limiti citati risulta immediato.

Riprendendo l'analisi delle problematiche della scienze sociali non può essere dimenticata la difficoltà di reperimento dei dati empirici che come abbiamo visto condiziona molte discipline. A ciò si aggiunga che molto spesso questi dati sono di tipo qualitativo piuttosto che quantitativo e che di norma queste informazioni sono di tipo privato, intimo e personale, espressione di fenomeni soggettivi unicamente accessibili al singolo essere umano. Questi fattori, assieme ad altri citati in precedenza, costituiscono alcuni dei limiti che impediscono l'adozione del metodo degli esperimenti di laboratorio anche in questo tipo di scienze. Altre limitazioni sono costituite dall'impossibilità di isolare i fenomeni sociali, difficilmente ripetibili, dal proprio contesto, dal proprio spazio e dal proprio tempo.

La debolezza delle basi empiriche su cui si fondano ha portato le scienze dell'uomo ad un uso eccessivo del linguaggio per la definizione della proprie teorie, accettando una visione essenzialistica della realtà. La ricerca dell'essenza rappresentata con una parola sottrae spesso risorse agli studiosi, preoccupati più dal capire e far capire i concetti che nello svilupparne di nuovi. Le limitazioni di questo orientamento trovano profondo riflesso nelle difficoltà del progresso di queste discipline.

Discipline che, nell'ambito delle scienze dell'uomo, nascono non dallo studio della realtà e dei suoi processi, ma da divisioni artificialmente introdotte degli scienziati per semplificare e per "atomizzare" gli aspetti di un unico ambito di studio. La visione multidisciplinare con la separazione dei fenomeni e la assegnazione a discipline diverse compromette la comprensione integrale di ognuno di essi, limitando il progredire della scienza nel suo complesso.

Se a ciò si aggiunge il profondo, e pericoloso, legame tra gli aspetti e gli interessi pratici con quelli teorici dell'opera dello studioso, portatore esso stesso di una propria individualità e soggettività, si ha un quadro sintetico delle motivazione che hanno, finora, impedito alle scienze dell'uomo di raggiungere i successi ottenuti da quelle della natura nello studio e nella comprensione della realtà.

---

preoccupantemente grande di comportamenti può essere razionalizzato come ottimale, comprovando la pericolosa e seducente flessibilità di tale euristica."

L'adozione del metodo proposto dalla simulazione se da un lato apre prospettive nuove per l'intero panorama delle scienze, dall'altro promette di rappresentare una autentica rivoluzione soprattutto per quelle dell'uomo; gran parte dei punti di debolezza, citati in precedenza, che hanno afflitto il procedere del sapere di queste ultime potrebbero venire risolti attraverso l'applicazione rigorosa di questa metodologia di studio.

Se per le scienze di natura, forti del loro consolidato metodo sperimentale, l'utilizzo della simulazione potrebbe essere utile per avere una rappresentazione delle teorie, per studiare fenomeni non riproducibili in laboratorio e per l'analisi dei sistemi complessi, limitandosi a dare un contributo al procedere della conoscenze; per quelle dell'uomo l'introduzione della simulazione rischia di avere un impatto pari a quello della rivoluzione gallileiana.

Una simile affermazione si giustifica se si pensa che grazie al nuovo strumento è possibile quella integrazione tra teorie e dati empirici, da sempre principale problema per gli studiosi di queste discipline, necessaria per fare raggiungere alle scienze dell'uomo il rango di scienze mature. La simulazioni coincidono con le teorie che vengono così espresse in forma esplicita, precisa e rigorosa. Una teoria definita in questo modo fornisce inevitabilmente una serie di predizioni empiriche che coincidono con i risultati prodotti dalla applicazione della simulazione.

Si comprende come, per le scienze dell'uomo, l'adozione delle simulazione come modalità di studio possa rappresentare un progresso importante quanto quello compiuto dalle scienze della natura con il metodo sperimentale. La mancata applicazione delle metodologie degli esperimenti di laboratorio, da sempre preclusi agli studiosi del sociale a causa di limiti intrinseci all'oggetto di studio, ha sempre costituito un limite nello studio dei fatti empirici. Le simulazioni sono in grado di far cadere questi limiti, fornendo agli studiosi del sociale, una serie di strumenti in grado riprodurre fenomeni complessi determinati da molte cause, valutandone le potenziali interazioni, senza dover prescindere dal contesto di riferimento. La realtà simulata può essere facilmente controllata all'interno del laboratorio sperimentale virtuale che la simulazione mette a disposizione anche nello studio di fenomeni sociali, impossibili da trattare diversamente.

La capacità delle simulazioni di rappresentare e ricreare sistemi complessi, che costruiscono la regola nella tipologia degli oggetti di studio delle scienze umane, costituisce un ulteriore elemento in grado di confermare la validità del metodo proposto che presenta altri indiscussi vantaggi.

Primo dei quali quello di rendere preciso e oggettivo il linguaggio delle scienze dell'uomo, le cui teorie, tranne poche eccezioni hanno carattere qualitativo. Le simulazioni consentono un notevole progresso alle scienze dell'uomo grazie alla quantificazione di teorie e dati empirici: da un lato viene reso oggettivo il momento della derivazione di predizioni dalle teorie e dall'altro viene reso certo il momento della loro verifica. Le teorie assumono caratteri di oggettività senza le implicazioni nascoste dovute all'uso del lin-

guaggio. Ma la simulazione non prescinde dai vantaggi forniti dall'uso del linguaggio: coniuga la potenza descrittiva delle espressioni qualitative con gli aspetti quantitativi della teoria incorporata nella realtà della simulazione, annullando ogni rischio di essenzialismo del linguaggio.

Il linguaggio della simulazione può diventare, in questo modo, lo stesso linguaggio della scienza e la simulazione può assumere il ruolo di metodo unico e uniforme applicabile a tutte le discipline dell'uomo. Diventando uno strumento comune a tutti gli studiosi di scienze umane, la simulazione può superare le artificiose divisioni introdotte dalla suddivisione multidisciplinare di un unico oggetto di studio, trasformando tante osservazioni parziali degli studiosi in una visione unitaria e integrata dei fenomeni. La divisione verticale della realtà[17], per categorie di fenomeni, viene soppiantata dal nuovo modo che le simulazioni hanno di suddividere la realtà: in fasce orizzontali. Le categorie e i fenomeni che compongono la realtà sono suddivise in fasce orizzontali, partendo dal basso[18], da quelle più semplificate. Una simile suddivisione favorisce lo studio delle interazioni tra i fenomeni, allargando le capacità di analisi e sintesi degli studiosi.

L'adozione di un unico metodo dovrebbe abbattere le tre grandi separazioni artificiose (tra mente e natura, tra individuo e società e tra la visione sincronica[19] e quella diacronica[20]) che hanno diviso la realtà oggetto di studio delle scienze dell'uomo, dando un corpus dottrinale unitario alla comprensione dei fenomeni umani.

2.5 Ambito di applicabilità e obiettivi dei modelli simulativi

Riprendendo la ripartizione proposta da Axtell (2000), il metodo simulativo può avere tre ambiti distinti di applicabilità in relazione con l'utilizzo congiunto delle altre tecniche di rappresentazione e comprensione del reale:

- può rappresentare un semplice strumento in grado di rappresentare i risultati ottenuti per via analitica: in questo caso la realtà può essere perfettamente modellizzata in forma matematica e la simulazione può intervenire, da un lato, nella fase di controllo-verifica dei risultati e, dall'altro, per la descrizione delle predizioni empiriche del modello;

- può costituire un valido metodo complementare alle tecniche di formalizzazione matematica: la simulazione interviene ad analizzare il

[17] Funzionale per le scienze della natura, ma spesso inefficace in quelle dell'uomo in virtù della incapacità a valutare le interazioni tra i fenomeni.

[18] Seguendo l'impostazione *bottom-up* in contrasto con quella tradizionale *top-down*.

[19] Visione indipendente dal tempo, non storica, dei fenomeni.

[20] Visione storica che tiene conto del ruolo svolto dal tempo nel determinare i fenomeni.

comportamento dinamico di un modello solo parzialmente risolvibile con le tecniche analitiche;

- è in grado di rappresentare uno strumento completamente alternativo alle tecniche di analisi tradizionali: è questo il caso di tutti quei fenomeni, ed in particolar modo dei sistemi complessi, che per loro natura si prestano in maniera non sempre efficace ad essere analizzati con gli strumenti canonici adottati dalla scienza; per questi fenomeni l'unico metodo di studio può essere rappresentato dalla simulazione.

Identificati gli ambiti di applicazione del modello simulativo, risultano evidenti gli obiettivi del suo utilizzo come metodo di ricerca scientifica[21].

La simulazione è uno strumento in grado di permettere al ricercatore di:

1. ricreare la realtà dei fenomeni oggetto di studio in laboratori sperimentali virtuali, formalizzando la teoria ipotizzata dallo studioso;

2. verificare la validità il modello proposto, valutando coerenza interna ed esterna delle assunzioni alla base; in questo caso è possibile distinguere tre tipologie di validità che possono essere soddisfatte:

   - *validità strutturale*: si ha quando il modello non si limita a fornire gli stessi risultati che derivano dalla dinamica del sistema osservato, ma ne riproduce fedelmente anche il comportamento;
   - *validità replicativa*: garantisce la capacità della simulazione di riprodurre con i propri risultati i dati già osservati del fenomeno reale;
   - *validità previsionale*: fornisce allo studioso la possibilità di valutare comportamenti dei fenomeni in istanti successivi sulla base di risultati forniti dalla simulazione;

3. esplorare la realtà alla ricerca di relazioni tra i fattori alla base del fenomeno; in questa fase è concessa la possibilità allo studioso di influenzare parametri e comportamenti delle variabili in cerca di interazioni ancora ignote;

4. prevedere la dinamica del modello, e di riflesso della realtà in esso ricreata, sulla base delle condizioni date, grazie alla capacità computazionale delle informazioni contenute.

[21] Si prescinde, qui, dagli obiettivi perseguiti dagli usi della simulazione per scopi diversi da quelli scientifici che possono ridursi al seguente elenco:

- come mezzo per agire sulla realtà;
- per creare *robot* ed altri artefatti fisici;
- come metodo per la divulgazione e l'apprendimento;
- come strumento per il gioco, il divertimento e la creatività artistica.

2.6 Simulazione ad agenti in contesti economici

Chiariti gli aspetti teorici del metodo simulativi, valutata la sua applicabilità alle scienze sociali non resta che valutarne le implicazioni dell'utilizzo nell'ambito della scienza economica con particolare riferimento alle tecniche di modellizzazione dal basso, in primis quelle *basate su agenti*.

> "Our point of departure in agent-based modelling is the individual: We give agents rules of behavior and then spin the system forward in time and see what macroscopic social structures emerge."(Epstein and Axtell, 1996).

Gettando un ponte tra il pensiero di Hayek[22] e i concetti dell'elaborazione automatica decentrata, Lavoie, Beatjer e Tulloh affermavano nel 1990, con felice intuizione, le considerazioni che seguono:

> "(...) lo scopo sarebbe quello di stabilire i vincoli, specificare l'ambiente istituzionale o le regole di decisione per gli agenti e poi eseguire la simulazione per vedere cosa accade. L'idea non è quella di creare un modello matematico che ha proprie conclusioni implicite nelle premesse. Invece è quella di eseguire la simulazione come un esperimento mentale, dove ciò che è interessante non sono tanto i risultati finali, ma il modo in cui il processo funziona. E noi, i programmatori, non sapremo come il processo finirà per risultare fino a che non avremo eseguito l'esperimento mentale. L'ordine dovrebbe emergere non dal disegno del programmatore, ma dall'interazione spontanea delle parti componenti."

A distanza di poco più di un decennio, nonostante i successi ottenuti[23] non abbiano ancora costituito una massa critica sufficiente tale da far riconoscere il metodo al di fuori della cerchia degli specialisti, la formulazione condizionale che caratterizza il brano lascia il posto a certezze e in esso si ritrova gran parte del paradigma della simulazione, con espliciti riferimenti ai *modelli basati su agenti*[24]

Questo tipo di modelli, grazie al loro notevole grado di flessibilità, costituisce una delle tecniche di simulazione maggiormente indicate nello studio

[22] Ispirato dalla ipotesi di ordine spontaneo e dagli schemi del *laissez-faire* economico.

[23] Frutto della sensibile crescita del numero di applicazioni, in ambito sociale, sviluppate negli ultimi anni secondo questa metodologia grazie anche ai marcati progressi nel calcolo computazionale ed alla diffusione di strumenti informatici realizzati ad hoc per questi scopi. A questo riferimento si veda il capitolo IV in cui viene presentato il progetto *Swarm*, nato come linguaggio comune per lo sviluppo di simulazioni in ambito economico e non.

[24] In seguito citati con acronimo generale come ABM, *Agent Based Model* o nella accezione più specifica di Tesfatsion (2002), in ambito economico, come ACE, *Agent-based Computational Economics*.

dei fenomeni sociali. La concezione teorica di un simile strumento deriva dalla convinzione di poter "spiegare", in termini scientifici, come la dinamica di sistemi complessi sia il risultato di interazioni fra *agenti*, il cui comportamento deriva da regole di base estremamente semplici.

Un sistema sociale all'interno di una simulazione ABM è rappresentato e modellato come un insieme di entità autonome, definite appunto *agenti*. Ogni elemento del sistema ha, da un lato, proprie caratteristiche, definite da variabili, che ne definiscono lo stato, e dall'altro, proprie regole di azione in grado di farne modificare lo stato in istanti diversi. Come le caratteristiche, anche le regole di comportamento possono essere modificate in seguito all'evoluzione dell'ambiente e all'interazione con altri agenti.

Il mondo della simulazione è quindi un ambiente artificiale estremamente fluido e vivo in perfetta analogia con quello reale. La sua struttura non è, e non può, essere definita ex-ante, ma è nota solo ex-post a causa dei meccanismi di retroazione che caratterizzano il comportamento degli agenti. Superato il dualismo proposto dallo schema induzione-deduzione, con i modelli ad agenti è possibile cogliere la sintesi di fenomeni semplici, spiegando l'emergenza spontanea di regolarità partire dal basso. Ciò è profondamente diverso dalla logica tradizionale predominante nell'ambito di scienze come la sociologia o l'economia dove, seguendo l'impostazione top-down, le regolarità aggregate rappresentano il frutto di meccanismi artificiali di coordinamento, definiti ex-ante.

L'obiettivo dello studioso non è più quello di svelare questi meccanismi, ma diventa quello di identificare le regole, proprie di ciascun agente introdotto nell'ambiente, in grado di riprodurre le strutture dei fenomeni osservati. La capacità del modello di ricreare la realtà rappresenta il successo nell'identificazione di tali regole, facendo del modello stesso una possibile spiegazione del fenomeno.

Si supera in questo modo la logica duale tra micro e macrostruttura: i modelli ad agenti consentono lo studio congiunto del tutto e delle parti che lo compongono, il modo in cui interagiscono e si influenzano reciprocamente.

L'apparato metodologico e concettuale così definito, frutto anche di strumenti propri di altri campi di ricerca[25], ha l'indubbio pregio di consentire allo studioso di perseguire i seguenti risultati (Axelrod, 1997):

- elaborare teorie integrate ed omnicomprensive derivanti dall'interazione di agenti adattivi;
- verificare tali teorie attraverso l'analisi sperimentale dei risultati ottenuti, utilizzando anche i tradizionali strumenti dell'analisi matematico-statistica;

[25] Per citare qualche esempio si pensi alle reti neurali, agli algoritmi genetici, agli automi cellulari, etc.

- estendere e migliorare il modello proposto a realtà diverse da quelle inizialmente osservate.

Le caratteristiche dei modelli proposti, risultato della loro tecnica costruttiva e della loro natura dinamica, rendono inapplicabile uno dei cardini del metodo scientifico tradizionale: la replicabilità dei risultati ottenuti, con profonde implicazioni in termini di robustezza all'errore del modello stesso. L'unica possibilità di verifica delle teoria incorporata nella simulazione è rappresentato dall'esecuzione ripetuta di quest'ultima, tenendo conto delle possibilità da parte dello studioso di intervenire su parametri o condizioni iniziali.

Particolarmente importante per il successo in questa attività di verifica risulta la capacità di utilizzare tecniche appropriate per la costruzione degli agenti, delle loro regole di comportamento e dell'ambiente in cui si svolgono le loro azioni e interazioni. Questi aspetti verranno sinteticamente approfonditi nei paragrafi seguenti.

2.6.1 Principali aspetti nella definizione degli agenti, attivi nella simulazione

Rifacendosi alle nozioni introdotte da Wooldridge e Jennings (1995) è possibile individuare una serie di proprietà fondamentali che caratterizzano la definizione degli agenti, in linea con il metodo ABM.

Secondo questa classificazione, un agente definito in questo modo ha una propria *autonomia*, essendo in grado di controllare stato e azioni che lo caratterizzano, ed una propria *capacità di interazione*, espressione della abilità nel coordinare il proprio comportamento con quello degli altri agenti. L'agente non solo è in grado di *reagire* in risposta a variazioni dell'ambiente, ma è in grado di assumere un comportamento *propositivo*, finalizzato al raggiungimento di un dato obiettivo.

Prescindendo dalle tecniche utilizzate, per la creazione di un simile agente è necessaria la capacità di dotare ognuno di questi soggetti di funzionalità di base in grado di gestire aspetti di acquisizione, elaborazione e trasmissione delle informazioni. L'elaborazione del flusso informativo proveniente dall'agente stesso e dall'ambiente di riferimento, costituito da altri agenti, ne determina il comportamento, attivo o propositivo nel senso prima illustrato. La capacità di valutazione dell'effetto delle proprie azioni costituisce un ulteriore elemento caratterizzante l'agente, capace nei modelli *Cross-Target* (Terna, 1994) di sviluppare forme di coerenza interna.

L'analogia con caratteristiche e comportamenti degli esseri umani ha portato gli studiosi ad identificare simili agenti come "sistemi intelligenti artificiali", frutto di concetti e strumenti elaborati sulla base dell'idea di riprodurre artificialmente alcuni aspetti dell'intelligenza umana.

Questi concetti, elaborati per la prima volta negli anni '50, da Turing e von Neumann hanno trovato rapido sviluppo grazie ai progressi dell'in-

formatica e sono stati alla base dell'evoluzione di tre distinti paradigmi di studio dai confini spesso labili:

- l'**Intelligenza Artificiale (IA)**: consente la realizzazione di sistemi artificiali in grado di produrre comportamenti intelligenti[26] sulla base di una concezione simbolica dell'intelligenza come attività razionale volta al raggiungimento di un determinato scopo; un esempio di modelli costruiti con queste tecniche è rappresentato dai *sistemi esperti*;

- la **Vita Artificiale (ALife)**; sviluppatasi negli anni '80 grazie all'opera di Langton, questa impostazione ha avuto come obiettivo iniziale[27] quello di riprodurre processi naturali di tipo biologico, quali l'evoluzione[28] e la vita stessa. La capacità di sistemi definiti secondo questo paradigma di comportamenti intelligenti deriva da meccanismi di selezione artificiale dei sistemi migliori, più adatti, da parte dell'ambiente, in linea con l'evoluzione dei fenomeni naturali. Strumenti messi a disposizione da questa impostazione per la costruzione di agenti sono principalmente gli *automi cellulari*, i *algoritmi genetici* ed i *classifier system*;

- il **Connessionismo**: se le tecniche dell'Intelligenza Artificiale prescindono dai meccanismi di funzionamento delle strutture umane impegnate nella ricezione, elaborazione e trasmissione dell'informazione finalizzata all'azione, gli studiosi del paradigma connessionista fanno di questo aspetto il fulcro dei loro modelli; secondo Parisi (1989), i primi emulano l'intelligenza mentre i secondi la simulano cercando di riprodurne gli aspetti del funzionamento, facendo emergere comportamenti "intelligenti" grazie a processi di apprendimento ripetuto; tipico prodotto di questa impostazione è rappresentata dalle *Reti Neurali Artificiali (RNA)*.

Attualmente, la comunità scientifica degli studiosi sembra aver adottato una netta separazione solo tra l'Intelligenza Artificiale, utilizzata in ambito informatico per la ricerca di strutture logiche[29], e gli altri due paradigmi di studio. Connessionismo e Vita Artificiale hanno trovato notevoli punti di contatto, grazie all'apporto di contributi interdisciplinari di una variegata comunità di studio costituita da ricercatori sia di scienze della natura sia di quelle dell'uomo.

[26] In questo ambito è possibile individuare due distinte impostazioni: una forte secondo la quale i modelli proposti sono in grado di replicare la "mente" umana ed una debole secondo cui attraverso questi strumenti è solo possibile una analisi dei processi cognitivi.

[27] Definito dallo stesso Langton (1992) quello di sviluppare un paradigma computazionale sulla base di processi che caratterizzano gli esseri umani.

[28] Prodotto di selezione e variabilità, in linea con l'impostazione darwiniana.

[29] Secondo Langhton (1992), volta piuttosto alla realizzazione di "soluzioni intelligenti" che di autentici comportamenti razionali, troppo condizionata dagli scopi pratici assunti dall'informatica.

2.6.2 Considerazioni metodologiche sul grado di capacità cognitiva degli agenti

Funzionale alla scelta della tecnica di costruzione degli agenti è l'obiettivo perseguito con la simulazione e del tipo di fenomeno che costituisce l'oggetto di studio. Un aspetto importante che deve essere attentamente valutato dallo sviluppatore di questi modelli riguarda il grado di intelligenza da attribuire agli agenti con notevoli implicazioni sul livello di complessità del modello.

Il ricercatore si trova di fronte ad un *trade-off*, tra semplice e complesso, con notevoli conseguenze non solo sul grado di realismo della simulazione e sulla capacità di rappresentazione del reale, ma anche sul binomio efficacia-efficienza dell'intero processo simulativo.

La letteratura in materia sembra essersi divisa con una netta contrapposizione di posizioni tra i principali esponenti della comunità scientifica impegnata nella ricerca con l'uso del metodo della simulazione.

A portare una schematizzazione delle varie possibilità a disposizione degli studiosi è intervenuta la recente classificazione di Terna (2001) che suddivide le simulazioni utilizzando come determinanti, da un lato, il livello cognitivo attribuito agli agenti e, dall'altro, il grado di strutturazione dell'ambiente. Gli agenti si distinguono sulla base del livello cognitivo in: *minded*[30] e *no minded*. I primi sono rappresentati da entità caratterizzate da ridotta capacità di percepire ed elaborare le informazioni provenienti dall'ambiente e da regole di condotta fisse e predeterminate; i secondi, invece, grazie alle proprie capacità cognitive, sono in grado di elaborare regole di comportamento ed azioni sulla base delle informazioni percepite. Questi agenti operano in ambienti che possono essere: strutturati o non strutturati. In questa accezione, per ambiente strutturato si intende un ambiente in cui intervengono meccanismi di coordinamento tra le azioni degli agenti a differenza dell'ambiente non strutturato dove tali meccanismi sono assenti. Sulla base di questa suddivisione è possibile classificare le simulazioni ABM in:

1. modelli con agenti *no minded* operanti in un ambiente non strutturato;
2. modelli con agenti *minded* inseriti in un ambiente non strutturato;
3. modelli con agenti *no minded* che operano in un ambiente strutturato;
4. modelli con agenti *minded* inseriti in un ambiente strutturato.

Grazie a questo quadro di riferimento è possibile catalogare l'intera opera dei principali studiosi della materia, polarizzata intorno alle posizioni di Epstein e Axtell secondo il paradigma di agenti BDI (*Beliefs, Intentions, Desires*), da un lato, e di Axelrod, dall'altro.

[30] Anche detti *cognitivi*.

Per i primi, l'utilizzo di agenti BDI, complessi, sofisticati e dotati di un elevato grado di capacità cognitiva, costituisce un valido apporto al realismo della simulazione, capace di ricreare più fedelmente la realtà complessa. Una simile impostazione comporta per gli studiosi la necessità di avere una piena conoscenza dei processi e delle caratteristiche delle entità che costituiscono l'oggetto di studio. Il realismo della simulazione comporta un prezzo elevato in grado di risorse impegnate nel suo sviluppo, ma potrebbe essere necessario soprattutto nell'ambito dei fenomeni delle scienze dell'uomo, in virtù degli schemi mentali complessi che caratterizzano l'azione degli esseri umani e la loro capacità di risposta all'ambiente. Di questo parere Chattoe (1998) che afferma che un utilizzo di agenti *no mind* all'interno di modelli evolutivi porta a spiegazioni non plausibili dei fenomeni rappresentati.

Di diverso avviso, i ricercatori che ritengano che i modelli ABM non debbano essere finalizzati alla riproduzione *sic et simpliciter* della realtà oggetto di studio, ma alla comprensione processi fondamentali che sono alla base dei fenomeni più meno complessi. In linea con questa impostazione, espressa da un punto di vista teorico da Axelrod (1997), mentre il fenomeno può essere complesso, le ipotesi implicite nel modello di spiegazione della realtà e nelle regole di costruzioni degli agenti[31] devono restare semplici. La complessità del fenomeno è frutto dell'interazione di questi agenti eterogenei tra loro e con l'ambiente e non dell'impostazione data al modello e agli agenti. Il grado di semplicità del modello è direttamente proporzionale al livello di comprensione del fenomeno e della realtà da parte degli studiosi, dotati di limitate capacità cognitive e quindi maggiormente in grado di capire le assunzioni alla base e gli effetti prodotti.

2.6.3 Principali tecniche di rappresentazione del processo cognitivo degli agenti

Definito il livello di capacità cognitiva attribuito al singolo agente, esistono alcuni strumenti teorizzati all'interno delle scienze cognitive citate in precedenza che consentono la rappresentazione dell'intero processo.

Sistemi esperti

Sono algoritmi che, concretizzandosi in programmi, permettono di riprodurre e replicare compiti e azioni di esseri umani con specifiche competenze. Attraverso questi strumenti, definiti nell'ambito delle tecniche di *problem solving*, è possibile esplicitare regole e schemi comportamentali alla base di particolari azioni umane. Sistemi costruiti secondo questa logica sono perfettamente in grado di fornire risposte a stimoli predefiniti provenienti dall'ambiente, ma presentano uno scarso livello in termini propriamente

[31] Secondo il principio KISS (*Keep It Simple, Studid*) proposto ironicamente dallo stesso Axelrod (1997).

cognitivi. Raggiungono ottimi risultati nella loro sfera di competenza, ma sono privi di meccanismi cognitivi di apprendimento che consentano loro di accrescere il patrimonio di conoscenze e la capacità di dare risposte al di fuori del proprio ambito specifico.

Modelli di questo tipo, dotati di una struttura modulare, devono essere caratterizzati da una attenta e sistematica fase di acquisizione dei dati provenienti dall'ambiente. Elaborato il flusso informativo, il sistema esperto deve essere in grado di applicare schemi inferenziali capaci di far individuare l'azione da intraprendere. Una volta identificata la regola d'azione, il sistema deve essere in grado, controllando la propria logica interna, di fornire una giustificazione del processo decisionale adottato.

Automi cellulari (AC)

Nell'ambito del paradigma della Vita Artificiale e dei sistemi complessi proposto da Langton, gli automi cellulari sono perfettamente integrati e forniscono un esempio di applicazione ricorsiva ad una struttura di un semplice insieme di regole. In breve,

> "Un automa cellulare consiste di un reticolo regolare di *automi finiti*, che sono i più semplici modelli formali di macchina. Un automa finito, in ciascun punto del tempo, può trovarsi in uno solo di un numero finito di stati, e le sue transazioni nel tempo da uno stato all'altro sono governate da una "tabella di transizione": dati un certo ingresso ed un certo stato interno, la tabella di transizione specifica lo stato che deve essere assunto dall'automa finito nell'intervallo di tempo successivo. In un automa cellulare, l'input necessario proviene dallo stato che deve essere assunto dall'automa finito nell'intervallo di tempo successivo. In un automa cellulare, l'input necessario proviene dallo stato degli automi adiacenti del reticolo. In tal modo, lo stato di un automa al tempo *t+1* è una funzione degli stati dell'automa stesso e dei suoi vicini immediati al tempo *t*. Tutti gli automi del reticolo obbediscono alla stessa tabella di transizione e ogni automa cambia stato nello stesso istante, intervallo dopo intervallo. Gli automi cellulari sono buoni esempi del tipo di paradigma computazionale che fa al caso della vita artificiale: una determinazione dal basso del comportamento, parallela e locale." (Langton, 1992)

In definitiva, le caratteristiche essenziali di un automa cellulare sono brevemente:

- il suo "stato" che può essere un numero o una proprietà diversi per ciascuna "cella";

- il suo "vicinato", ossia l'insieme delle celle con cui interagisce;
- il suo "programma", ossia il set di regole che definiscono i cambiamenti che l'automa deve assumere in risposta al suo stato attuale e a quello dei suoi "vicini". [32]

Definiti in questo modo, gli automi sono in grado di darsi una "auto-organizzazione" (*self-organization*) e di assumere comportamenti simili a quelli della vita (*life-like behavior*).

Gli AC furono studiati dalla fine degli anni '40 da Wiener, Zuse, Ulam e soprattutto da von Neumann che fu uno dei primi ad intuire le potenzialità dello strumento e, con la sua opera, fu il primo a coglierne l'essenza della vita artificiale.

Dopo anni di oblio, intorno all'inizio degli anni '70, gli AC tornarono prepotentemente alla ribalta del panorama scientifico grazie all'opera di Conway, capace di ideare un gioco matematico (*Life*), divenuto molto popolare.

Solo all'inizio degli anni '80, gli AC sono stati integrati con la teoria dei sistemi complessi grazie all'opera di Wolfram. Lo studio di quest'ultimo dimostrò come anche sistemi molto semplici sono in grado di generare una elevata varietà di comportamenti, permettendo una classificazione degli AC in quattro classi:

- classi 1 e 2 (ordinate): partendo da una configurazione casuale di celle, queste dopo un numero più o meno elevato di interazioni si cristallizzano in stati fissi (classe 1) o di breve periodo (classe 2). La perturbazione di una cella viene rapidamente riassorbita.
- classe 3 (caotica): una configurazione iniziale si mantiene casuale, manifestando disordine e mancanza di struttura. Una perturbazione si propaga rapidamente su tutto l'automa, senza mutare l'impressione visiva generale di mancanza di una struttura riconoscibile.
- classe 4 (complessa): dopo un periodo transitorio più lungo che nelle classi ordinate, si sviluppano delle strutture periodiche di forma varia, fisse o mobili, che possono interagire tra loro in vari modi, annichilendosi, assorbendosi o trasformandosi. Le perturbazioni possono propagarsi oppure arrestarsi.

Sulla base di questa classificazione è anche possibile analizzare il contenuto informativo degli AC. In virtù della loro imprevedibile risposta alle perturbazioni, gli automi complessi sono quelli che manifestano un contenuto informativo più elevato. Mentre negli altri due casi esso risulta basso a causa della loro "mancata" risposta agli stimoli esterni.

[32] Volendo estendere una analogia con la biologia, è possibile identificare questo set di regole come il suo "genotipo".

Algoritmi genetici

In perfetta analogia con il principio biologico dell'evoluzione, gli studiosi della vita artificiale hanno saputo ricrearne i processi nell'ambito dei fenomeni cognitivi.

Nati dall'intuizione di Holland, alla ricerca di una soluzione efficiente nei problemi matematici di ottimizzazione delle funzioni, si basano sul concetto di sopravvivenza delle "soluzioni migliori", o meglio di quelle che si adattano meglio alla natura del processo. Definiti da Goldberg (1989) come:

> "(...) algoritmi di ricerca basati su meccanismi di selezione naturale e sulla genetica. Integrano un principio di sopravvivenza della struttura maggiormente adatta all'ambiente con un meccanismo di scambio di informazioni, strutturato, ma stocastico, per formare un algoritmo di ricerca che presenta un po' dell'intuito innovativo insito nella ricerca umana. Ad ogni generazione, un nuovo insieme di creature artificiali (sequenze) è generato sulla base degli individui migliori dell'insieme precedente (...)".

Per descrivere questo processo è possibile riferirsi all'opera di Koza (1990) secondo cui:

> "Gli algoritmi genetici sono algoritmi matematici ad alto parallelismo che trasformano popolazioni di oggetti matematici individuali (in genere sequenze binarie di lunghezza fissa) in popolazioni utilizzando operatori mutuati dalle operazioni genetiche naturali come la riproduzione sessuata (*cross-over*) e della riproduzione proporzionale al grado di adattamento (principio di adattamento del migliore di Darwin).
>
> Gli algoritmi genetici partono da una popolazione iniziale di individui (generata casualmente) e iterativamente valutano il grado di adattamento all'ambiente di ogni individuo della popolazione e applicano operatori genetici su diversi individui della popolazione per generare una nuova popolazione."

La sequenza tipica alla base dell'algoritmo prevede le seguenti fasi:

a) generazione della popolazione;

b) valutazione degli individui;

c) riproduzione;

d) applicazione degli operatori "genetici";

e) ripetizione iterativa del ciclo per un numero definito di generazioni.

I principi fondamentali dell'evoluzione come metodo di ricerca euristico possono essere sintetizzati nella maniera seguente.

I focus di ricerca, le soluzioni proposte, sono rappresentate come individui. Dal momento che è possibile costituire una popolazione di individui, l'evoluzione è un metodo di ricerca multifocale.

Il criterio di ottimizzazione deve essere monodimensionale ed è rappresentato dal grado di adattabilità (*fitness*) dell'individuo.

A questa funzione di *fitness* possono essere assegnati vincoli che rappresentano termini addizionali di penalizzazione.

Le nuove soluzioni candiate, chiamate soluzioni "figlie" (*offsprings*), sono create usando i membri attuali della popolazione, chiamati "genitori" (*parents*).

Le due operazioni di evoluzione della popolazione sono la *mutazione* e la ricombinazione (*crossover*). La mutazione comporta che il figlio abbia le stesse proprietà dei propri singoli genitori con alcune variazioni marginali. Mentre la ricombinazione, il *crossover*, implica che le proprietà del figlio siano derivate da quelle dei due genitori.

La selezione dei genitori è casuale, ma vi sono influenze nella preferenza per quelli in grado di adattamento maggiore in termini di *fitness*: questo implica che individui con adattabilità più elevata abbiano più figli.

Per ogni nuovo figlio inserito nella simulazione, un altro elemento della popolazione deve essere rimosso al fine di mantenere costante la numerosità dell'ambiente. Questo processo di selezione può essere compiuta in maniera del tutto casuale o seguendo nuovamente la *fitness* di ogni elemento. In modo particolare, lo scenario ipotizzato nel primo caso implica che la durata media di vita sia uguale per tutti gli elementi della simulazione; per contro nel secondo caso prevale una concezione per cui sono premiati i membri in grado di adattarsi meglio, e quindi di sopravvivere più a lungo.

Classifier system

Rifacendosi alla definizione data da Goldberg (1989), un *classifier system* è "un sistema di apprendimento automatico che impara regole sintatticamente semplici, denominate *classifier*, per guidare le sue azioni in un ambiente arbitrario."

Sin dalla definizione si può intuire come questi sistemi rappresentino una evoluzione dei sistemi esperti basati su regole. A differenziare i due strumenti intervengono una serie di caratteristiche peculiari dei *classifier*: il parallelismo nella generazione delle regole, l'introduzione di un sistema competitivo di valutazione delle stesse e soprattutto l'utilizzo degli algoritmi genetici per la creazione di queste ultime. L'utilizzo degli algoritmi genetici permette il superamento dei limiti mostrati dai sistemi esperti in termini di definizione di una ampia numerosità di regole di funzionamento e in termini di eccessiva complicatezza della sintassi di queste istruzioni.

Per la definizione di questo strumento è possibile riferirsi a Margarita (1992) che individua i seguenti come elementi fondamentali dei *classifier*:

- una popolazione di regole, generata in maniera casuale o in modo da incorporare regole definite a priori, che costituisce un insieme di stringhe[33] a lunghezza fissa;
- un sistema di "recettori" capaci di ricevere gli *input* dall'ambiente, codificare le informazioni ricevute e convertire tali *input*, mediante le rappresentazioni interne di regole, in opportune azioni sull'ambiente compiute (*output*) dai *classifier* "attivati";
- un sistema di valutazione, "ad asta", che determina, sulla base di un meccanismo di attribuzione di crediti/debiti[34] quale dei *classifier* attivati agisca effettivamente[35].
- procedure e strumenti contabili in grado di aggiornare il patrimonio dei singoli *classifier* in base ai crediti/debiti maturati in seguito alle decisioni realizzate;
- un algoritmo genetico che genera nuove regole in sostituzione di quelle non più adeguate in risposta a *input* che non corrispondano ad alcuno dei *classifier* a disposizione del sistema.

Lo strumento definito in questo modo evidenzia capacità di adattamento che dipendono dalla creazione di *classifier* semplici ed eterogenei, adattati a rispondere a *input* specifici.

Nell'ambito di modelli ad agenti un simile strumento è stato utilizzato per descrivere processi cognitivi sulla base dell'osservazione del reale, ipotizzando che ogni individuo, di fronte ad un problema, possa disporre di schemi risolutivi ritenuti praticabili e frutto dell'esperienza.

Una applicazione particolarmente interessante di questa impostazione in ambito economico è rappresentata dal modello ASM, *Artificial Stock Market*.

[33] Definite in alfabeto ternario e in grado di rappresentare una opportuna condizione e la relativa azione, secondo una logica IF-THEN.

[34] Basato sull'algoritmo ideato dallo stesso Holland e denominato *bucket brigade algorithm* che comporta per ogni regola attivata la diminuzione del proprio patrimonio di crediti di un costo iniziale. Successivamente la regola viene valutata con il confronto con l'ambiente e sulla base di questa valutazione il *classifier* attivato vede il proprio patrimonio crescere di un certo premio, funzionale a tutte le regole che hanno concorso a creare l'azione. L'iterazione del processo consente alle regole migliori di accrescere il proprio patrimonio di crediti, e contemporaneamente di depauperare quelle meno efficaci, creando un sistema di regole in continua evoluzione sulla base di processi progressivamente più efficienti. Una ulteriore distinzione interviene nella scelta del *classifier* da attivare che nell'impostazione determinista sarà sempre quello che dispone del patrimonio più elevato mentre in quella stocastica dipenderà da una funzione di probabilità proporzionale al proprio patrimonio.

[35] Un *classifier* può agire direttamente sull'ambiente, producendo un'azione verso l'esterno, oppure attivarne un altro.

Reti Neurali Artificiali (RNA)

Diffuse nell'ambito delle teorie del connettivismo, le RNA rappresentano uno speciale tipo di elaborazione delle informazioni. Una RNA consiste di celle primitive (*cells*) che lavorano in parallelo e sono collegate tra di loro attraverso *link*, o connessioni, di tipo diretto. Il principio fondamentale di elaborazione di queste unità è la distribuzione di *pattern* di attivazione attraverso le connessioni. Questo processo presenta una perfetta analogia con quanto avviene nei meccanismi di base del cervello umano, dove l'elaborazione delle informazioni avviene grazie al trasferimento degli impulsi di attivazione da un gruppo di neuroni agli altri attraverso le *sinapsi*. Ad un simile meccanismo di elaborazione delle informazioni è anche conosciuto come *parallel distribuited processing* (PDP).

Alla base della scelta di ispirarsi ai processi cerebrali nello sviluppo di algoritmi di elaborazione della informazioni c'è l'alto livello in termini di risultati ottenuti in compiti cognitivi altamente complessi come, per esempio, il riconoscimento di *pattern* visivi o uditivi. Proprio la difficoltà di questi compiti, insieme agli alti risultati raggiunti, hanno costituito uno stimolo per gli studiosi nell'approfondimento dei meccanismi alla base del funzionamento del cervello umano e da questi studi sono nati gli algoritmi RNA che derivano il proprio nome da questa motivazione storica. Attualmente, la maggior parte delle architetture RNA non cerca più di imitare il processo biologico ispiratore, ma può essere considerata semplicemente come una classe di algoritmi di elaborazione parallela.

In Terna (1995):

> "Una funzione a rete neurale produce un vettore di *output* da un vettore di *input*, sulla base di un insieme di parametri (detti pesi) la cui determinazione non si scosta 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. (...) La rete deve produrre in uscita un valore predefinito per ogni set di dati in input; tali valori di output sono contenuti nell'insieme di dati su cui si fonda l'apprendimento (*training set*); la rete stessa, con cui si definiscono i pesi w, sarà verificata su un insieme di dati di controllo (*validation set*) e quindi applicata a dati via, via nuovi. Gli *output* su cui la rete apprende o comunque contenuti nel *validation set* sono anche indicati con il nome di *target*."

In questi modelli, l'informazione cognitiva è solitamente distribuita attraverso la rete ed è immagazzinata nella struttura topologica dei pesi delle connessioni. Le reti sono organizzate da metodi di apprendimento automatici, che semplificano notevolmente lo sviluppo di specifiche applicazioni.

Il risultato di questi meccanismi fornisce soluzioni costituite da conclusioni plausibili (*best match*) che soppiantano i risultati in termini di logica classica delle soluzioni di intelligenza artificiale (*exact match*). Ciò costituisce un grande vantaggio in tutte le situazioni e in tutti i fenomeni in cui non può essere individuato un chiaro insieme di regole logiche alla base. Inoltre la plausibilità delle soluzioni trovate costituisce un ulteriore indubbio vantaggio in termini di tolleranza allerrore. Un esempio di tale robustezza può essere fornito dalla capacità di tolleranza di una RNA in caso di rumore (*noise*) e di perturbazioni dei dati forniti in ingresso alla rete: al crescere del rumore o della perturbazione, solitamente la qualità dei risultati diminuisce solo lentamente.

Uno dei maggiori vantaggi della reti neurali è la loro abilità nel generalizzare. Questo significa che una rete opportunamente addestrata può essere in grado di elaborare informazioni[36] che non ha mai conosciuto in precedenza. Nella mondo reale, gli sviluppatori di sistemi di questo tipo non dispongono quasi mai dell'intera serie di dati che caratterizzano il fenomeno osservato: normalmente solo una piccola parte di tutti i possibili casi è disponibile per la costruzione della rete neurale. Per raggiungere la massima generalizzazione, l'insieme dei dati disponibili deve essere opportunamente diviso in tre parti:

- il **training set** che è usato per addestrare la rete (*in sample*); l'errore calcolato su questi dati campione è minimizzato durante l'apprendimento;
- il **validation set**, utilizzato per calcolare i risultati ottenuti della rete su casi su cui non è avvenuto l'apprendimento durante la fase precedente (*out of sample* o fuori campione);
- un **test set** per il controllo complessivo dei risultati della rete.

La fase di apprendimento dovrebbe concludersi una volta raggiunto il minimo errore nella verifica del *validation set*, punto in cui la capacità di generalizzazione della rete è massima. Se l'apprendimento non viene interrotto, possono verificarsi fenomeni di "sovrapprendimento" che che comportano un decadimento delle prestazioni della rete, nonostante l'errore calcolato sul *training set* resti ancora contenuto. Al termine della fase di apprendimento, la rete dovrebbe essere sottoposta al controllo del terzo insieme di dati, il *test set*, per una verifica complessiva delle prestazioni.

Una rete è composta di celle e connessioni dirette e "pesate" tra le unità. In analogia con l'attivazione che passa nei neuroni della biologia, ogni cella riceve un *input* dalla rete che deriva dagli *output* dalle unità con cui è in collegamento.

[36] A patto che siano dello stesso tipo di quelle utilizzata in fase di apprendimento.

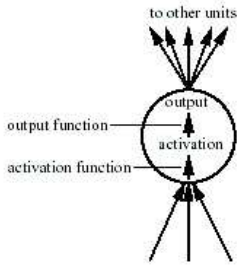

Figura 2.1: Lo schema di un nodo di una RNA

In alcuni modelli, il processo di elaborazione delle informazioni all'interno dell'unità avviene grazie ad apposite *funzioni di attivazione* e *funzioni di output*. La prime elaborano l'*input* delle unità dai valori, opportunamente pesati, che costituiscono il prodotto di celle precedenti. Successivamente viene elaborata una nuova attivazione sulla base di quanto entra in *input* nell'unità; in seguito questo risultato è elaborato dalla funzione di *output* per generare il prodotto in uscita dall'unità.

Sulla base delle funzioni svolte all'interno della rete, è possibile distinguere tre tipi di celle:

- le unità le cui attivazioni sono gli *input* del problema sono chiamate *nodi di input*;
- quelle che invece rappresentano il risultato prodotto dalla rete sono definite *nodi di output*;
- esiste poi una terza tipologia di nodi che va a costituire nell'architettura di una rete uno strato nascosto, non visibile dall'esterno, e queste celle sono dette *nodi hidden*.

In molti modelli a reti neurali, il tipo dei nodi dipende dalla relativa posizione topologica occupata dall'unità nella rete: se una cella non ha connessioni in entrata, ma solo in uscita allora sarà un nodo di *input*, mentre se lunità è priva di legami in uscita, ma dispone solo di *link* in entrata allora sarà un nodo di *output*. La presenza simultanea di connessioni in entrata e in uscita qualificherà la cella come un nodo *hidden*.

La possibilità di creare strutture complesse e annidate di reti neurali rende possibile che il risultato di una cella interna sia trattato come *output* della rete.

Una nuova attivazione è calcolata dal risultato della unità precedente, solitamente moltiplicata per il *peso* della connessione che collega questa unità con quella corrente, per la precedente attivazione e per il suo *bias*[37]. La sua

[37]L'introduzione del bias, solitamente rappresentato dal peso costante pari ad 1 applicato all'input o al hidden, ha il pregio di arricchire i gradi di libertà della funzione.

formula generale è la seguente:

$$a_j(t+1) = f_{act}(net_j(t), a_j(t), \theta_j)$$

dove
$a_j(t)$ valore di attivazione dell'unità j al tempo t
$net_j(t)$ input della rete dell'unità j al tempo t
θ_j valore soglia (bias) dell'unità j

La funzione di attivazione più comune è la *logistica* che calcola gli *input* della rete semplicemente sommando tutte le attivazioni opportunamente pesate e successivamente normalizza il risultato grazie, appunto, alla funzione logistica $f_{act}(x) = 1/(1 + e^{-1})$. La nuova attivazione al tempo $(t + 1)$ è compresa tra $[0, 1]$[38].

L'input della rete $net_j(t)$ è calcolato con la seguente formula:

$$net_j(t) = \sum_i w_{ij} o_i(t)$$

che conduce alla nota funzione di attivazione logistica

$$a_j(t+1) = \frac{1}{1 + e^{-(\sum_i w_{ij} o_i(t) - \theta_j)}}$$

dove
$a_j(t)$ è l'attivazione dell'unità j al tempo t
$net_j(t)$ input della rete nell'unità j al tempo t
$o_i(t)$ output dell'unità i al tempo t
j indice per l'unità della rete
i indice dell'unità precedente dell'unità j
w_{ij} peso del collegamento dall'unità i all'unità j
θ_j valore soglia (bias) dell'unità j

Il passo successivo del processo prevede l'intervento della funzione di *output* che calcola il risultato di ogni unità dal relativo valore di attivazione. Questa funzione rende possibile elaborare l'attivazione per ottenere l'*output*. Molto spesso questa funzione è costituita dalla semplice identità:

$$o_j(t) = f_{out}(a_j(t))$$

dove
$a_j(t)$ è l'attivazione dell'unità j al tempo t
$o_i(t)$ output dell'unità i al tempo t

[38] A rigore matematico tra $]0, 1[$, ma i valori tra 0 e 1 sono ottenuti a causa di imprecisioni aritmetiche.

j indice per l'unità della rete

Molto spesso si preferisce che il prodotto dell'unità sia compreso tra [0,1] e allora il valore di attivazione viene normalizzato con una funzione come la seguente:

$$o_j(t) = \begin{cases} 0 & \text{se } a_j(t) < 0 \\ 1 & \text{se } a_j(t) > 1 \\ a_j(t) & \forall \in]0, 1[\end{cases}$$

La direzione delle connessioni mostra la direzione del passaggio dell'attivazione nella struttura. L'unità da cui parte una connessione è chiamata *unità sorgente* mentre quella a cui arriva il *link* è chiamata *unità target*.

Ad ogni connessione è assegnato un peso che ha il compito di definire l'effetto dell'*output* di una unità su quella che riceve l'attivazione. Se il peso è negativo, la connessione è inibitoria, mentre, se è positivo, ha una funzione stimolatrice; ciò in analogia con i meccanismi biologici.

La strutture a reti neurali più utilizzate sono costruite gerarchicamente dal basso verso l'alto: l'*input* si propaga nelle unità solo dalle celle degli strati che le precedono. Per il loro flusso unidirezionale dell'informazione all'interno della struttura, senza retroazioni, queste reti sono definite *feed-forward*.

Per poter propagare i nuovi valori di attivazione nelle unità, l'algoritmo deve poter accedere a tutte le celle[39] utilizzando un ordine sequenziale, che definisce le modalità di aggiornamento della rete. Sono cinque i principali tipi di aggiornamento delle attivazioni all'interno della rete:

- *sincrono*: le unità modificano le loro attivazioni tutte insieme dopo ogni passo; dopo che tutte le unità hanno assegnato il nuovo valore di attivazione si passa al calcolo del nuovo *output* dell'unità;
- *permutazione casuale*: le unità calcolano le loro nuove attivazioni e le funzioni di *output* in maniera sequenziale; l'ordine è definito casualmente, ma ogni unità è selezionata solamente una volta in ogni passo.
- *causale*: l'ordine è definito da un generatore di numeri casuali; questo non garantisce che tutte le unità siano selezionate una sola volta in ogni passo di aggiornamento;
- *in serie*: l'ordine è definito sulla base del numero attribuito ad ogni cella all'interno della struttura;
- *topologico*: le unità sono aggiornate sulla base della loro posizione nella topologia della rete. Questo ordine corrisponde alla modalità biologica

[39] Identificate da un numero all'interno della struttura della rete.

di propagazione dell'attività da *input* ad *output* ed è particolarmente utilizzato nelle reti *feed-forward*

Un aspetto particolarmente importante nell'ambito di ricerca e di sviluppo di RNA riguarda le modalità di correzione dei pesi per ottenere il comportamento desiderato dal sistema. Un metodo di aggiustamento dei pesi molto diffuso è quello basato sulla regola di Hebbian, che ipotizza che la connessione tra due unità sia tanto più forte se entrambe le unità sono attivate allo stesso momento.

La regola di Hebbian può essere formalizzata come:

$$\Delta w_{ij} = g(a_j(t), t_j)h(o_i(t), w_{ij})$$

dove
w_{ij} peso del collegamento dall'unità i allunità j
$a_j(t)$ è l'attivazione dell'unità j al tempo t
t_j è l'input target dell'azione, il valore di output desiderato dalla cella j
$o_i(t)$ output dell'unità i al tempo t
$g(\ldots)$ funzione che dipende dall'attivazione dell'unità e dell'input di apprendimento
$h(\ldots)$ funzione che dipende dall'output del precedente elemento e dal peso corrente della connessione

Il *training* di una rete neurale *feed-forward* con i meccanismi di apprendimento descritti in precedenza consiste nella procedura seguente:

- un *pattern di input* è sottoposto alla rete;
- l'*input* è propagato in avanti nella rete fino a quando l'attivazione raggiunge lo strato di nodi di *output*.
- i risultati ottenuti nel punto precedente sono confrontati con quelli desiderati come *target*. L'errore, la differenza δ_j tra il risultato prodotto o_j e il valore desiderato dal nodo di output j come *target*, è utilizzato insieme al prodotto o_i dall'unità sorgente per calcolare gli opportuni aggiustamenti al peso della connessione w_{ij}. Per calcolare le correzioni necessarie per le unità interne (tipicamente quelle di *hidden*) per cui non si dispone di un set di valori obiettivo desiderati, si utilizzano i valori di errore degli strati successivi, già calcolati grazie ad un meccanismo di retropropagazione dell'errore, chiamato *backward propagation*.

  Nell'apprendimento immediato (*online*), le correzioni dei pesi Δw_{ij} sono applicate alla rete dopo ogni *pattern* di *training*, mentre nell'apprendimento *offline* i cambiamenti necessari nei pesi sono cumulati per tutti i *pattern* di addestramento e gli aggiustamenti sono applicati al termine dell'intera sequenza di tali *pattern*.

Attualmente, il più famoso algoritmo che segue questo schema è la *backpropagation*. Utilizzando questa modalità di correzione dei pesi, l'apprendimento *online* è significativamente più rapido se confrontato con quello *offline*, soprattutto in caso di ampi set di *training* con molti casi simili.

La regola di correzione dei pesi basata sull'algoritmo della *backpropagation* può essere così formalizzata:

$$\Delta w_i j = \eta \delta_j o_i$$

$$\delta_j = \begin{cases} f'_j(net_j)(t_j - o_j) & \text{se il nodo } j \text{ è una unità di output} \\ f'_j(net_j) \sum_k \delta_k w_{jk} & \text{se il nodo } j \text{ è una unità nascosta} \end{cases}$$

dove
η costante che rappresenta il fattore di apprendimento
δ_j errore, la differenza tra l'output ottenuto e quello desiderato come target, per l'unità j
t_j l'input di apprendimento dell'unità j
o_j rappresenta l'output della precedente unità i
i indice di una unità precedente a quella corrente con peso w_{ij} da j a j
j indice dell'unità corrente
k indice di una unità successiva a quella corrente con peso w_{jk} da j a k

2.6.4 La costruzione della simulazione e del proprio ambiente: lo schema ERA

Nell'ambito delle scelte legate alla costruzione degli oggetti che rappresentano gli oggetti all'interno delle simulazioni, una interessante proposta arriva da Terna (1998, 2000): l'utilizzo dello schema ERA, *Environment-Rules-Agents*, particolarmente efficace con le tecniche di programmazione *ad oggetti*[40].

Secondo questa impostazione, nella costruzione del modello e degli agenti è necessario seguire quattro diversi livelli:

1. il primo strato fornisce una rappresentazione dell'ambiente, il contesto, in cui gli agenti sono destinati ad operare;

2. il secondo livello è quello specifico degli agenti, definibili come istanze, esemplari derivanti da una classe-tipo;

3. il terzo strato è quello in cui avviene la gestione delle modalità con cui gli agenti decidono le loro azioni e può essere considerato come la rappresentazione delle capacità cognitive del singolo agente; nello schema, l'agente interroga un oggetto superiore gerarchicamente, a

[40] Argomento trattato nel successivo capitolo.

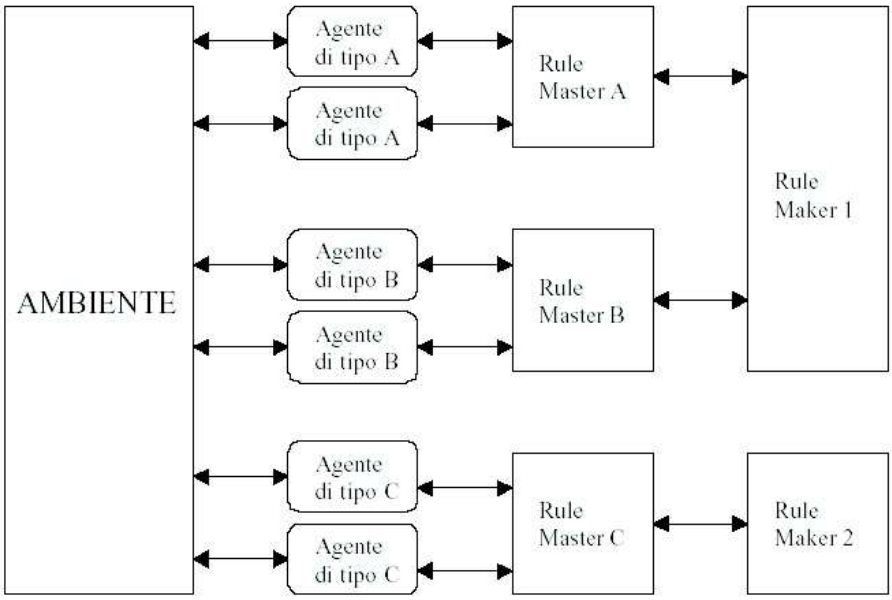

Figura 2.2: Lo schema ERA

cui è demandata la funzione di gestione delle regole[41] (*Rulemaster*): l'oggetto, ricevuti dall'agente i dati necessari, fornirà la regola specifica da applicare;

4. il quarto strato concerne la costruzione delle regole: allo stesso modo con cui gli agenti interrogano i loro gestori di regole, questi ultimi interrogheranno un nuovo oggetto, il generatore di regole (detto *Rulemaker*), per modificare nel tempo i propri schemi di azione[42]; il generatore di regole rappresenta le diverse strutture cognitive degli agenti.

Da un punto di vista teorico, il principio fondamentale del modello proposto è il mantenimento su due livelli concettuali distinti di ambiente, che rappresenta il contesto per mezzo di regole e dati generali, e agenti, con le loro strutture di dati interne.

Per semplificare il codice della simulazione, si assume che gli agenti non comunichino tra di loro in maniera diretta, ma che i flussi informativi passino sempre attraverso l'ambiente.

Un simile schema ha, inoltre, l'indubbio pregio di rendere rigorosa la scrittura del codice informatico della simulazione, rendendone modulare il

[41] Regole che potranno essere semplici o complesse a seconda se l'agente sia *minded* o *no minded*.

[42] In perfetta analogia con quanto avviene nel mondo reale in cui gli individui adeguano i propri comportamenti in risposta all'esperienza maturata e alle conoscenze acquisite.

processo di costruzione. In particolar modo la modularità è apprezzabile nella definizione dei comportamenti attribuiti ai singoli agenti, grazie alla possibilità di sostituzione degli algoritmi[43] utilizzati per i gestori e i generatori di regole: ciò implica poter modificare il tipo di agente, operando solo su singole componenti del modello senza necessariamente dover rivedere l'intera struttura.

2.7 I modelli ad agenti nelle scienze economiche: l'*Agent-based Computational Economics*

Secondo il paradigma proposto per lo studio dei fenomeni appena descritto, le moderne economie di mercato decentrate costituiscono sistemi adattivi complessi[44]. Questi sistemi sono costituiti da una numerosità di agenti coinvolti in interazioni locali parallele. Le interazioni locali danno vita alle regolarità a livello macroeconomico come i protocolli del mercato condiviso e le norme di comportamento che attraverso un meccanismo di retroazione (*feed-back*) influenzano la determinazione delle stesse interazioni locali. Il risultato è costituito da un complesso sistema dinamico di eventi periodici concatenati e causali che connettono gli effetti di comportamenti individuali e delle interazioni reticolari tra questi.

Questo intricato feedback a due vie tra micro e macro-struttura è stato individuato all'interno dei sistemi economici da lungo tempo. Tuttavia, agli economisti sono sempre mancati i mezzi per includere nei loro modelli queste relazioni in maniera quantitativa nella loro piena complessità dinamica. La caratteristica più importante dei modelli economici quantitativi espressi in termini micro è sempre stata la loro costruzione top-down. In questi modelli una grande importanza è sempre stata attribuita su meccanismi di coordinamento esogeno delle azioni come le regole di decisioni prefissate, le assunzioni in termini di conoscenze comuni, l'introduzione di agenti rappresentativi e il vincolo imposto di equilibrio di mercato. L'assenza ruolo solitamente lasciato alle interazioni personali costituisce un ulteriore ed eccessiva semplificazione per agenti sin troppo limitati.

Fortunatamente, in maniera lenta, ma decisa, i progressi negli strumenti di modellizzazione hanno esteso le possibilità per gli studiosi dell'economia.

[43] Passando, ad esempio, da RNA a *classifier system* o algoritmi genetici.

[44] Un sistema adattivo complesso può essere definito come un sistema composto da una numerosità di elementi indipendenti, anche diversi tra loro, dotati di capacità di agire secondo modalità non sempre prevedibili e capaci di influenzare reciprocamente le proprie azioni. Il comportamento finale di un sistema così definito emerge dalle interazioni dei singoli elementi, dando vita in maniera endogena ad un'autorganizzazione spontanea. Al risultato finale del sistema concorrono solo le azioni dei singoli agenti. Sono definiti adattivi in forza del fatto che tali sistemi sono in grado di modificarsi ed adattarsi in risposta alle variazioni dell'ambiente, presentando un comportamento attivo.

I ricercatori possono attualmente creare modelli in termini quantitativi di una grande varietà di fenomeni economici complessi.

Un ramo di questo lavoro è stato conosciuto come economia computazionale basata ad agenti, o secondo l'acronimo inglese ACE. ACE rappresenta lo studio computazionale dei sistemi economici, modellati come sistemi in evoluzione, costituiti da interazioni di agenti autonomi.

I ricercatori che adottano questa metodologia di indagine e rappresentazione del reale fanno affidamento sui laboratori computazionali per studiare l'evoluzione delle economie di mercato decentrate in opportune e controllate condizioni sperimentali. Sono due gli interessi che guidano questi studi:

1. il primo di tipo descrittivo, si concentra sulle spiegazioni implicite dei comportamenti emergenti a livello aggregato;

2. il secondo di tipo normativo, è attento agli schemi dei meccanismi del fenomeno, per poter influire su di esso.

Come in un esperimento di laboratorio, gli studiosi iniziano costruendo una economia con una popolazione iniziale di agenti. Questi potranno rappresentare sia soggetti economici in senso stretto (per esempio: un consumatore, un produttore o un processo produttivo[45], una istituzione finanziaria, un operatore del mercato azionario, etc.) sia un soggetto che rappresenti vari altri fenomeni sociali o ambientali (come per esempio l'azione di governo, del clima, etc.). I ricercatori identificheranno lo stato *ab initio* del sistema economico sulla base dei valori attribuiti inizialmente agli agenti. I parametri iniziali di ciascuno di essi potranno tener conto delle caratteristiche del tipo, di regole di comportamento interiorizzate, di modelli di comportamento interni (tra i quali le modalità di apprendere e di comunicare) e le informazioni accumulate internamente sul proprio stato e su quello degli altri agenti. L'economia così creata evolvere nel tempo senza alcuna necessità di intervento da parte del ricercatore: tutti gli eventi accadono in maniera sequenziale derivano dalla linea temporale delle interazioni tra gli agenti, senza che nessun meccanismo di controllo esterno sia permesso.

Questo apparato metodologico è lo stesso dei ricercatori sulla vita artificiale[46]. Entrambi i gruppi di studiosi condividono lo scopo di dimostrare

[45]Come esempio può essere citata l'applicazione del metodo jVE (java Virtual Enterprise) definito in Terna (2001) con cui vengono realizzate simulazioni capaci di "far funzionare" l'azienda simulata e non solo di rappresentarla in modo animato sulla base di sequenze predeterminate di eventi. Con questo metodo i processi aziendali vengono rappresentati tramite sequenze di codici numerici (le sequenze complete formano le "ricette") ciascuno dei quali costituisce un passo nella realizzazione di un bene, merce o servizio che sia. Le singole unità produttive possono essere autonomi micro-meccanismi operanti nel sistema economico, oppure unità integrate all'interno di imprese.

[46]Seguendo l'originale definizione inglese *artificial life*, o più semplicemente ALife; tecnica illustrata nel paragrafo precedente.

maniera implicita come regolarità globali possano svilupparsi partendo dal basso (impostazione bottom up), in seguito a ripetute interazioni locali di agenti autonomi. I modelli computazionali sono utilizzati da entrambi i gruppi sia per scopi descrittivi sia normativi al fine di sviluppare teorie coerenti ed omnicomprensive dei fenomeni.

A differenziare le due impostazioni è la concezione teorica alla base della costruzione dei modelli: per i ricercatori ALife, fedeli alla definizione forte di intelligenza artificiale, identificano nei propri modelli la sintesi della vita reale riprodotta con l'elaboratore eletrronico, mentre gli studiosi ACE vedono nelle proprie simulazioni la rappresentazione di processi economici esistenti o potenziali piuttosto che fenomeni reali dotati di un intrinseco interesse. La linea di demarcazione tra le due discipline, tuttavia, sembra sempre più assottigliarsi.

Capitolo 3

Il linguaggio della simulazione

Resi evidenti i benefici dell'uso della simulazione nell'ambio delle scienze *"dell'uomo"*, così definite in Parisi (2001), l'osservatore di questi fenomeni necessita di adeguati strumenti per realizzare e sottoporre a verifica il proprio modello. Durante queste fasi del processo cognitivo possono essere superati alcuni rigidità del rigore matematico e possono essere colte intuizioni in grado di interpretare in maniera del tutto originale i fenomeni del mondo reale.

Per poter aver successo in questa delicata serie di operazioni lo sperimentatore deve poter disporre di strumenti che consentano alla simulazione, in maniera semplice ed efficace, di soddisfare requisiti in termini di leggibilità e replicabilità dei risultati. L'importanza di questi requisiti diventa evidente se si pensa alla necessità di consentire anche ad altri studiosi di capire, replicare ed eventualmente raffinare i modelli oggetto di sperimentazione, che devono diventare patrimonio dell'intera comunità scientifica.

Il patrimonio di conoscenze personali e la variegata disponibilità di mezzi forniti dall'informatica hanno costituito allo stesso tempo vincoli ed opportunità per gli sperimentatori, indirizzati e, in alcuni casi, costretti ad utilizzare programmi e linguaggi di programmazione diversi tra loro. Se da un lato questo ha favorito creatività e capacità di individuare di volta in volta lo strumento migliore per compiere i processi simulativi, dall'altro il proliferare dei linguaggi di programmazione (dapprima Fortran, Pascal e C, successivamente C++, Java e VisualBasic) ha presto costituito un enorme vincolo per la diffusione dei risultati ottenuti.

Per ovviare a queste difficoltà, e per dotare la comunità scientifica di una sorta di linguaggio comune, il Santa Fe Institute ha fornito alla comunità scientifica una sorta di linguaggio comune, concreta risposta in grado di superare le difficoltà incontrate dagli studiosi nelle loro analisi e simulazioni di sistemi complessi con agenti eterogenei come quelli sociali.

Un gruppo di ricercatori dell'istituto americano, sotto la guida di Chris Langton, nel 1994 ha dato inizio al progetto Swarm con lo scopo di sviluppare sia un protocollo sia una serie di strumenti informatici standardizzati per lo sviluppo di modelli di simulazione multiagente. Secondo le intenzioni degli autori, Swarm doveva rappresentare per la comunità scientifica impegnata nell'*Agent Based Modelling* un opportuno ambiente di lavoro in grado di consentire ai ricercatori di focalizzare i propri sforzi nella effettiva implementazione dei modelli, evitando loro le complicazioni ed i limiti dei singoli linguaggi di programmazione. Attraverso l'uso di Swarm, la comunità scientifica avrebbe dovuto avere un unico, potente e flessibile strumento di lavoro standardizzato in grado anche di affrancare il lavoro degli utilizzatori da tutta quella serie di problematiche informatiche tipiche dell'uso dei vari linguaggi di programmazione. Si pensi, solo per citare un esempio, alla gestione dell'interfaccia grafica, la cui implementazione aveva rappresentato da sempre elemento in grado di sottrarre risorse all'effettiva attività di ricerca. Il risultato dell'attività del gruppo di lavoro guidato inizialmente da Langton si è concretizzato nell'insieme di librerie software scritte in Tkl/Tcl ed Objective C e raccolte con lo scopo di fornire un valido sostegno nell'attività di modellizzazione.

Sin dalla prima versione beta di Swarm rilasciata nel 1995 e dalla successiva versione stabile (la 1.0) del gennaio di due anni dopo, basate sulle architetture UNIX/Solaris e Linux a livello di sistema operativo, è apparsa chiara la necessità di rendere il più possibile compatibile l'insieme degli strumenti messi a disposizione dei ricercatori. In questa direzione sono andati gli sforzi del team di sviluppo impegnato nell'apertura dell'insieme di librerie anche verso altri sistemi operativi (quali DEC Alpha e Microsoft Windows). Espressione di questo impegno sono la versione 1.1 dell'aprile 1998 e la 2.0. La prima in grado di rendere possibile, attraverso l'uso del pacchetto software Cygwin Win32, anche a sistemi Windows 9.x/NT di utilizzare Swarm; la seconda di rendere possibile l'accesso alle librerie anche a programmatori Java. L'aver saputo mantenere le promesse e gli obiettivi prefissati ha garantito a Swarm una rapida diffusione rendendolo uno degli strumenti di simulazione più diffusi dalla comunità scientifica, estesasi progressivamente tra gli studiosi di scienze della natura e quelli delle scienze dell'uomo, dando vita ad interessanti approcci interdisciplinari.

Proprio la vivacità di questa comunità di ricerca, grazie alla modalità open source (con l'adesione alla licenza pubblica GNU) che caratterizza il progetto, ha saputo rapidamente apportare il proprio contributo alle librerie, fornendo continui sviluppi e costanti miglioramenti.

3.1 Principali caratteristiche di Swarm

Swarm è stato progettato per aiutare i ricercatori a costruire modelli in cui far interagire agenti elementari di basso livello, dando vita ad autentici "sistemi complessi". L'analisi a livello sistemico di questi modelli permette di far emergere pattern derivanti dall'azione comportamentale degli individui elementari.

La programmazione ad oggetti, per le proprie intrinseche caratteristiche, riesce particolarmente bene a rappresentare costruzioni di questo tipo ed è a questo che si è pensato nell'intera fase di concezione di Swarm.

Sin dalla sua ideazione il progetto Swarm doveva portare alla definizione di un insieme di librerie fondato su di un codice "orientato agli oggetti", in grado di favorire la scrittura e l'esecuzione di applicazioni complesse anche grazie alla disponibilità di numerosi strumenti accessori di aiuto.

L'esigenza di basare il linguaggio delle librerie su di un codice orientato agli oggetti ha comportato la scrittura di Swarm in una particolare estensione del linguaggio di programmazione C, l'Objective-C. La scelta ha reso possibile definire *"classi"* da cui creare istanze di individui, in linea con le metodologie tipiche della programmazione *"ad oggetti"* con tutti i benefici in termini di astrazione che questa comporta.

Le applicazioni così sviluppate hanno una forte struttura gerarchica che parte dal livello più elevato, quello dell'*observer Swarm* che definisce a sua volta il *model Swarm*, da cui vengono creati gli agenti individuali; successivamente vengono pianificati compiti e attività di questi ultimi e si procede alla raccolta delle informazioni sul loro stato. Il flusso informativo viene a sua volta messo a disposizione dei livelli superiori, rendendo possibile all'*observer Swarm* la diffusione, l'analisi e l'interpretazione dei dati immagazzinati.

In questa attività, sono a disposizione del ricercatore una preziosissima serie di strumenti che consentono una agevole gestione complesse quanto dispendiose attività informatiche che vanno dal controllo della memoria all'implementazione dell'interfaccia grafica, favorendone un significativo incremento della produttività.

3.2 La programmazione ad oggetti

Se i concetti fondamentali della programmazione ad oggetti erano già noti negli anni '40, a tal punto che molti informatici identificano le radici di queste tecniche di scrittura di applicazioni nell'introduzione delle subroutine, solo lo sviluppo dei moderni linguaggi ne ha reso possibile la grande diffusione.

Naturale evoluzione della programmazione strutturata, come è stata definita da J.Duntemann[1], questo concetto di programmazione consente di

[1] "La programmazione orientata agli oggetti è la strutturazione della programmazione strutturata."

sviluppare ed ottenere codice che funziona in modo molto diverso da quello scritto secondo i criteri procedurali tradizionali.

Queste differenze impongono una forma mentis completamente diversa da parte del programmatore, costretto a pensare contemporaneamente in termini di codice e dati che vengono a coincidere in solo *oggetto*. I vari oggetti vengono a combinarsi in reti strutturate che formano il programma completo, caratterizzato dall'interazione delle strutture elementari che lo compongono. Ogni componente del programma, identificabile come oggetto, risulta caratterizzato da uno stato (definito dai dati o *variabili istanza*), ma anche da un comportamento (rappresentato da funzioni in grado di operare sui dati o *metodi*) che costituiscono gli elementi di un unico concetto. In maniera più rigorosa, le funzioni costituiscono i metodi dell'oggetto e i campi della sua struttura di dati rappresentano le sue variabili di istanza.

Gli oggetti così definiti costituiscono, assieme alle relative modalità di interazione, gli elementi elementari del programma: dalla loro interconnessione deriva quella struttura reticolare con cui può essere rappresentata graficamente una applicazione definita secondo i canoni della programmazione ad oggetti. Ogni oggetto ha uno specifico compito da svolgere all'interno della struttura così definita ed è in grado di comunicare con gli altri oggetti attraverso opportune chiamate dette messaggi. E' attraverso questi ultimi che avviene la richiesta di esecuzione dei vari metodi.

In un programma possono, ovviamente, essere presenti contemporaneamente più oggetti di tipo diverso ed anche dello stesso tipo, ognuno dei quali può essere identificato come un componente autonomo e autosufficiente del programma. L'insieme degli oggetti dello stesso tipo forma una *classe*: tutti i membri di una classe sono in grado eseguire gli stessi metodi e sono caratterizzati dallo stesso insieme di variabili istanza. Ogni membro della classe, inoltre, viene definito una sola volta, attraverso ad una definizione comune. Quest'ultima operazione a livello classe se da un lato non genera alcun elemento utilizzabile dal programma, dall'altro genera una "tipizzazione" dell'oggetto che serve alla successiva costruzione di un insieme di oggetti simili, identificabili con le istanze della classe. Ognuna di queste istanze ha in comune variabili e metodi, ma risulta identificata da un proprio spazio di memoria in cui si trova allocata. Lo spazio di memoria viene ad essere l'elemento caratterizzante dell'istanza in grado di contenere il set di variabili peculiare dell'esempio stesso. Ogni oggetto ha, quindi, una zona di memoria per le proprie variabili. Dal processo di allocazione delle zone di memoria, invece, sono esclusi i metodi che, in quanto dotazione condivisa dall'intera classe, sono presenti solo ad un livello gerarchico più elevato.

Una simile concezione di programmazione offre al programmatore tre indiscussi vantaggi:

- la facilità di manutenzione del programma grazie ad una maggiore intelleggibilità del codice;

- una maggior facilità nel modificare le istruzioni del programma;
- la riutilizzabilità degli oggetti.

Ad essere indispensabile diventa una notevole capacità di astrazione che trova riflesso nella capacità di manipolare contemporaneamente dati e funzioni: dal lato dei dati, questi ultimi sono strutturati in unità più ampie che sono manipolate come singole entità; dal lato delle funzioni, i comportamenti possono essere incapsulati in metodi che possono essere riutilizzati anche senza essere riscritti in righe di codice. Una volta che l'oggetto sarà stato definito sarà sufficiente effettuare una chiamata per riutilizzarlo.

3.2.1 I vantaggi della programmazione ad oggetti

La scrittura di un programma secondo i canoni della programmazione ad oggetti prevede, come già evidenziato, una notevole capacità di astrazione. Il requisito è reso necessario dalla necessità di capire cause ed effetti tra gli eventi e di distinguere ciò che è importante da ciò che non lo è, evidenziando i legami funzionali tra le cose. Se questo modo di pensare è tipico dello scienziato-osservatore, lo è ancor più dello scienziato-sperimentatore che deve riprodurre le osservazioni nei propri programmi. La capacità di astrarre è garantita dai linguaggi di programmazione "ad oggetti", con particolare riferimento all'Objective-C che in questo presenta alcune differenze rispetto a Java, dalla separazione tra l'*interfaccia* e l'*implementazione*.

Nell'interfaccia si trovano le descrizioni delle funzionalità dei metodi di un classe mentre nell'implementazione si trovano le specifiche di come tali funzionalità debbano essere realizzate. Mentre l'interfaccia dichiara i metodi che l'oggetto può eseguire, l'implementazione può essere nascosta. Questa separazione netta tra interfaccia e implementazione serve a proteggere quest'ultima da azioni o accessi non voluti dell'utilizzatore e costituisce una delle proprietà fondamentali della programmazione ad oggetti che prende il nome di *incapsulamento*. Questa proprietà prevede che i valori delle variabili dentro un oggetto siano privati, resi inaccessibili al resto del programma, e vi siano appositi metodi scritti per "passare" tali informazioni (le variabili istanza e l'implementazione dei propri metodi) al di fuori dell'oggetto stesso.

Se da un lato questa proprietà può apparire limitativa dell'attività di programmazione, dall'altra grazie all'incapsulamento che si rende possibile la modifica dell'implementazione di un oggetto senza dover riscrivere anche i programmi che lo utilizzano.

Altrettanto importante, soprattutto nell'ambito della scrittura di modelli di simulazione basati su agenti, è un'altra caratteristica fondamentale della programmazione ad oggetti: l'*ereditarietà* ossia la capacità conferita ad ogni classe di ereditare tutte le variabili istanza e i metodi della classe da cui deriva. In questo modo è possibile definire una nuova classe (detta *sottoclasse*) da una classe già definita, (chiamata *superclasse*), adattandola

per particolari compiti sulla base di specifiche esigenze. La nuova classe sarà definita solo dai metodi e dalla variabili che differiscono dalla superclasse da cui eredita.

La interessante possibilità conferita ad ogni classe può essere usata come una superclasse, caratterizza i rapporti tra le classi in maniera fortemente gerarchica. Ogni classe può potenzialmente essere allo stesso tempo superclasse che sottoclasse per gruppi diversi di classi. Le complesse gerarchie di ereditarietà iniziano con una classe *"root"* (radice) senza superclassi da cui ereditano tutte le altre. Definita questa classe radice sarà possibile definire a catena le altre sottoclassi che avranno variabili e metodi di tutte le superclassi, gerarchicamente più elevate, da cui derivano.

L'ereditarietà si rivela particolarmente preziosa quando:

- si ha la necessità di estendere la definizione di una classe già definita per utilizzare nuovi metodi o variabili adattandoli per la specifica applicazione;
- occorre modificare il comportamento ereditato da un metodo;
- esiste l'esigenza di sostituire un metodo ereditato con uno nuovo.

La modifica o la sostituzione di un metodo è possibile scrivendo una classe che presenta tra i propri metodi uno con lo stesso nome di quello che si vuole sostituire che verrà quindi "oscurato". Il metodo originale rimarrà ancora valido per la superclasse e per tutte le sottoclassi che derivano direttamente da quest'ultima.

L'ereditarietà risulta proprietà fondamentale nella possibilità di riutilizzare il codice scritto per le varie applicazioni che può in questo modo essere condiviso senza necessità di essere ridefinito.

La capacità di soddisfare i requisiti di astrazione imposti dalla programmazione ad oggetti diventano evidenti e diventano ancor più apprezzabili nell'ambito della modellizzazione basata su agenti. Un agente del modello può essere definito come un oggetto "istanziato", tipo di un oggetto astratto iniziale, che eredita da questo delle proprietà generali in grado di qualificarlo e può al tempo stesso essere dotato di comportamenti e caratteristiche proprie.

A completare la caratteristiche peculiari che definiscono i canoni della programmazione ad oggetti occorre citare anche il *polimorfismo*, ossia la capacità di differenti oggetti di rispondere, ognuno secondo proprie modalità, a identici messaggi. Ciò è possibile grazie alla possibilità di definire nomi, per variabili e metodi, all'interno di una classe non in conflitto con quelli definiti esternamente. I nomi così definiti risultano validi solo a livello locale, o più correttamente, a livello "privato" della singola classe.

L'invio ad un oggetto di uno specifico messaggio comporta la richiesta di effettuare un determinato compito con l'indicazione puntuale del metodo

che dovrà essere eseguito. La possibilità data ad oggetti differenti di avere metodi con lo stesso nome determinerà un'interpretazione diversa dello stesso messaggio da parte dei vari oggetti destinatari quando questi abbiano metodi diversi che hanno lo stesso nome.

In questo modo è possibile tenere distinti o più precisamente astrarre il comportamento dell'oggetto dall'implementazione dei metodi, facendo sì che l'oggetto definito abbia accesso unicamente ai propri metodi senza essere influenzato da quelli con lo stesso nome eventualmente caratterizzanti altri oggetti.

La definizione dell'interfaccia risulta notevolmente semplificata, grazie alla possibilità di riutilizzare lo stesso nome per metodi con funzioni che svolgono compiti simili, ma che operano su differenti classi di oggetti.

Il codice che viene scritto diventa facilmente riutilizzabile ed estendibile anche grazie alla maggiore accessibilità alle funzioni che viene garantita enucleandole in più metodi. Eventuali nuove esigenze non previste in sede di scrittura del codice possono essere soddisfatte semplicemente aggiungendo una nuova classe con un nuovo metodo, lasciando le altre già eventualmente scritte e compilate.

Un ultimo, ma non per questo meno importante, aspetto fondamentale della programmazione orientata agli oggetti riguarda la gestione dell'allocazione delle zone di memoria. I limiti imposti dai linguaggi di programmazione procedurali cadono con l'introduzione dei canoni della programmazione ad oggetti. Utilizzando i primi il programmatore è costretto ad riunire tutte le parti del programma in un unico file reso eseguibile ed immodificabile al termine delle fasi del *"linking"* e della compilazione. L'intera quantità di memoria necessaria al programma viene allocata non appena questo viene lanciato, restando fissa ed immutabile durante l'intera fase di esecuzione dello stesso. Questo costituisce un limite evidente, rendendo la gestione della memoria demandata alle impostazioni del programmatore in sede di scrittura del codice piuttosto che dipendente dalle informazioni disponibili in fase di *runtime* (esecuzione).

Questo limite viene superato con l'utilizzo della programmazione ad oggetti con la quale è possibile realizzare programmi dinamici in grado di spostare parte delle scelte in termini di allocazione e gestione della memoria dalle fasi di linking e compilazione a quelle di esecuzione dello stesso. Questo garantisce all'utilizzatore del programma un più alto grado di autonomia durante la fase dell'esecuzione: vi saranno maggiori possibilità di lasciare decidere all'utilizzatore quali operazioni dovranno essere compiute dal programma, piuttosto che limitarne le azioni in maniera preordinata in sede di scrittura del codice, tenendo conto di limiti e necessità del compilatore.

A garantire il soddisfacimento di questi obiettivi interviene una nuova caratteristica peculiare della programmazione ad oggetti tipica dell'Objective-C: il dinamismo. Quest'ultimo può essere individuato nelle tre forme in cui si presenta:

- *dynamic typing* che fa si che venga attesa la fase di esecuzione per la determinazione della classe di un oggetto; questo consente di evitare di dover codificare l'oggetto in maniera statica;
- *dynamic binding* che determina nella fase di runtime quale metodo debba essere invocato;
- *dynamic loading* che aggiunge nuovi componenti al programma durante la sua esecuzione; eventuali altri moduli connessi al modulo principale possono essere mandati in esecuzione solo in risposta a specifiche azioni compiute dall'utilizzatore del programma, rendendo decisamente più efficiente ed efficace il processo di gestione della memoria.

3.2.2 Struttura dei programmi ad oggetti

I programmi scritti con linguaggi di programmazione orientati ad oggetti hanno due tipi di strutture: la prima può essere individuata nella struttura gerarchica della definizione delle classi determinata dall'ereditarietà, la seconda si rende evidente nelle sequenze di messaggi che vengono scambiati durante l'esecuzione del programma che formano una fitta rete di connessioni tra gli oggetti.

La gerarchia definita con l'ereditarietà fornisce informazioni su come gli oggetti siano legati per tipo.

La rete di connessioni tra gli oggetti specifica quale sia il funzionamento del programma.

I programmi scritti seguendo il paradigma della programmazione ad oggetti sono definiti in modo da spiegare la rete di oggetti con i relativi comportamenti e le sequenze dell'interazione tra gli stessi, e da disporre la gerarchia delle classi. La strutturazione è presente sia nella fase di esecuzione del programma che durante la sua definizione.

Una parte importante della scrittura del codice relativo riguarda proprio la disposizione della rete di oggetti che non può essere statica, ma viene modificata in maniera dinamica durante la fase di runtime. Le relazioni tra gli oggetti possono essere instaurate secondo le necessità e l'insieme di oggetti che svolgono un ruolo definito può essere modificato di volta in volta.

Alcune connessioni possono essere unicamente transitorie come avviene quando un messaggio può contenere un argomento che identifica un oggetto, per esempio il mittente, che viene passato al destinatario informandolo della possibilità di comunicare con lui. La risposta questo messaggio può favorire la creazione di una autentica catena di messaggi capace di determinare la struttura delle connessioni.

Ma non tutte le connessioni tra gli oggetti possono essere manipolate "al volo": alcune necessitano di essere registrate in strutture di dati di programma e per far questo sono disponibili varie modalità. Le connessioni tra gli

oggetti possono essere tenute in apposite tabelle o possono essere definiti appositi servizi che identificano gli oggetti sulla base del nome. Naturalmente la modalità più semplice è quella che prevede per ogni oggetto la possibilità di avere variabili istanza che mantengono traccia dei legami con gli altri oggetti con i quali comunica. Queste variabili, definite *outlet* dal momento che registrano l'uscita dei messaggi, definiscono le connessioni principali tra gli oggetti all'interno della struttura del programma.

Le connessioni tra gli outlet possono catturare molti tipi di diverse relazioni tra gli oggetti. Alcune volte la connessione avviene tra oggetti che comunicano ed agiscono in collaborazione, ognuno con il proprio ruolo da compiere e senza rapporti di subalternità con gli altri. Altre volte, invece, un oggetto potrebbe essere identificato come parte di un altro con tutti i rapporti di dipendenza che ne derivano.

La rete di oggetti viene attivata da stimoli esterni che possono essere di diversa provenienza a secondo del tipo di applicazione che è stata definita e a seconda degli obiettivi che si propone. Molto spesso i programmi scritti ad oggetti vengono mandati in esecuzione da sequenze di eventi che identificano tipi definiti di processi esterni. Si pensi al caso di un'applicazione con un interfaccia utente che è governata dagli eventi generati dall'utilizzatore attraverso l'uso dei dispositivi di input richiesti del programma (tipicamente attraverso una tastiera o un mouse).

Un altro aspetto importante nella scrittura del codice di un programma utilizzando un linguaggio ad oggetti è rappresentato dalla definizione delle classi. In particolar modo è fondamentale stabilire quando aggiungere una funzionalità ad una classe esistente aggiungendo una sottoclasse oppure quando definire una nuova classe interamente indipendente. Il problema può essere reso più chiaro immaginando i due casi estremi: da un lato un intera applicazione consistente in un solo oggetto, perdendo i vantaggi di polimorfismo ed ereditarietà, e dall'altro un programma formato da centinaia di oggetti differenti, ognuno con pochi metodi e limitate funzionalità.

Ovviamente è meglio evitare questi casi limite, cercano di mantenere oggetti sufficientemente ampi da svolgere una funzione sostanziale nel programma, ma allo stesso tempo sufficientemente limitati da mantenere un ruolo ben identificabile. Da questa scelta, funzionale agli obiettivi dell'applicazione, dipende la possibilità di comprendere interamente la struttura del programma.

Se l'oggetto deve essere usato in più circostanze, eventualmente anche in un altro progetto, è conveniente definirlo in una classe separata, aumentando il tasso di riutilizzabilità del codice. In linea generale, quest'ultima caratteristica della programma dovrebbe essere sfruttata al massimo, avendo cura di evitare di definire classi troppo estese da svolgere compiti così diversi che non possano essere adattate in altre situazioni. Se gli oggetti sono definiti come componenti, la loro riutilizzabilità è decisamente maggiore: ciò che funziona in un particolare sistema o in una specifica configurazione può

comportarsi altrettanto efficacemente in un'altra. Dividere le funzionalità tra diverse classi non complica necessariamente l'interfaccia del programma.

In analogia con quanto avviene nel mondo fisico per gli oggetti reali, anche quelli definiti all'interno di un programma combinano "stati" e "comportamenti": in questo modo è possibile immaginarli in termini di azioni compiute (cosa fanno), di modi di agire (come agiscono) e di interazione (come si connettono l'un l'altro). La scrittura di un programma orientato agli oggetti rappresenta la simulazione attraverso l'uso dell'elaboratore del funzionamento del fenomeno reale. Le reti di oggetti assomigliano e si comportano come modelli di sistemi reali: ogni componente del modello viene descritto in termini di comportamenti e delle sue interazioni con gli altri componenti. Dal momento che l'interfaccia del oggetto consiste nei suoi metodi, non nei suoi dati, è possibile iniziare il processo di progettazione pensando a cosa una componente del sistema farà, piuttosto che su come è rappresentata nei dati. Una volta che il comportamento di un oggetto è stato definito, viene scelta una opportuna struttura di dati e ciò riguarda la gestione dell'implementazione, non della progettazione iniziale. La scelta della struttura dei dati può essere modificata ripetutamente senza determinare cambiamenti nel progetto.

La realizzazione di un programma fondato su di un linguaggio di programmazione ad oggetti non richiede la scrittura di grandi quantità di codice. La possibilità di riutilizzare classi eventualmente definite in altri progetti o addirittura da altri sviluppatori all'interno della propria applicazione semplifica l'attività di scrittura del programma. Di pari passo con la crescita dell'insieme delle classi già definite, cresce la possibilità di riutilizzo di parti di codice da parte della comunità degli sviluppatori.

Le classi riusabili possono provenire da varie e disparate fonti e solitamente i linguaggi di programmazione orientata agli oggetti dispongono di numerose librerie con centinaia di classi utili per la realizzazione delle diverse applicazioni. Alcune di queste offrono funzioni di base, mentre altre sono più specifiche.

Solitamente, un gruppo di librerie di classi viene utilizzato per definire una parziale struttura del programma, un *framework* che può essere utilizzato per costruire una varietà di differenti tipi di applicazioni. L'uso di un simile framework, implica l'accettazione del modello del programma che mette a disposizione degli sviluppatori, e implica, l'adattamento della fase progettuale al modello stesso. E possibile usare framework per:

- inizializzare e modificare istanze delle classi del framework stesso;
- definire sottoclassi del framework;
- definire nuove classi ad hoc in grado di interagire con quelle del framework.

In questo modo parte della rete di oggetti del programma, con la relativa struttura gerarchica, è definita dal framework definito dalle librerie mentre il codice sviluppato dal programmatore personalizza e completa l'applicazione.

La programmazione orientata ad oggetti non solo consente le strutturazione dei programmi in un modo nuovo, ma allo stesso tempo aiuta nello sviluppo dei compiti che la costituiscono, ma consente anche di facilitare il compito nella definizioni di applicazioni sempre più complesse.

Proprio la complessità nello sviluppo di applicazioni favorisce un maggior grado di collaborazione tra i membri della comunità degli sviluppatori, riducendo al contempo la necessità di sottoporre a coordinamento gli sviluppi apportati da altri membri. La innegabile comodità di poter sviluppare applicazioni in vari moduli, sviluppabili singolarmente, determina significativi miglioramenti non solo nella intelleggibilità e riutilizzabilità del codice, ma anche nella fase di ricerca di eventuali errori (*debugging*), tutto ciò con indiscussi vantaggi in termini di affidabilità del codice stesso.

3.3 L'Objective-C

L'Objective-C è un linguaggio di programmazione ad oggetti basato sullo standard del C e introduce l'ambiente di sviluppo più avanzato disponibile ai tempi della nascita, l'ambiente OPENSTEP. Il linguaggio si è evoluto come un insieme di estensioni del C, mettendo a quest'ultimo le intere caratteristiche della programmazione orientata ad oggetti in maniera semplice e allo stesso tempo efficace. Queste estensioni sono numerose e derivano principalmente da Smaltalk, uno dei primi linguaggi di programmazione ad oggetti .

La sintassi dell'Objective-C rappresenta quindi una estensione del linguaggio C standard e il suo compilatore funziona sia con codice sorgente C (con piena compatibilità verso il C standard definito secondo gli standard ANSI) che Objective-C. In fase di compilazione i listati con i sorgenti in Objective-C sono caratterizzati dell'estensione ".m" a differenza di quelli scritti secondo lo standard C che hanno ".c".

Inoltre l'Objective-C può essere utilizzato come una estensione del C++, la più diffusa estensione ad oggetti del C, rendedo possibile allo sviluppatore di utilizzare congiuntamente gli aspetti positivi di tipici dei linguaggi citati, sfruttando i vantaggi dei vari linguaggi. Ciò consente ad esempio di scegliere quando realizzare una particolare funzionalità dell'applicazione secondo i canoni del paradigma ad oggetti e quando, per contro, seguire gli standard della programmazione procedurale per soddisfare la stessa esigenza.

Come evidenziato in precedenza, un oggetto associa i dati con le particolari operazioni che possono usare o modificare tali dati. Nel gergo dell'Objective-C, questi ultimi sono definiti come i *metodi dell'oggetto*, mentre i dati a cui si riferiscono costituiscono le sue variabili di istanza. Sostan-

zialmente un oggetto Objective-C coniuga una struttura dei dati (variabili di istanza) con un gruppo di procedure (metodi) in una singola unità di programmazione.

In Objective-C, le variabili di un'istanza di un oggetto sono contenute nell'oggetto stesso, rendendone possibile l'accesso allo stato solo attraverso ai propri metodi.

Il linguaggio presenta così i propri punti di forza che possono sintetizzarsi nella sostanziale facilità d'uso che si traduce in una chiarezza ed intelligibilità della sintassi impiegata nel codice. Ciò ha evidenti riflessi da un lato nella capacità di astrazione che trova nell'Objetive-C compiute possibilità di realizzazione e dall'altro nell'elevato grado di "dinamismo" che il linguaggio consente, superiore a quello disponibile con altre estensioni del C.

Proprio il dinamismo rappresenta uno dei cardini su cui si fonda il linguaggio grazie alla capacità di spostare gran parte delle decisioni in merito alla gestione e all'allocazione della memoria dalla fase della compilazione a quella dell'esecuzione. Ciò assicura un ampio grado di flessibilità nella scrittura del codice, incrementando notevolmente efficacia ed efficienza dell'applicazione.

Le uniche limitazioni accettate in termini di dinamismo scrivendo programmi in Objective-C come estensione del C++ derivano della mancanza delle proprietà del dynamic binding e di dynamic typing, descritte in precedenza.

3.3.1 Classi, oggetti e messaggi dell'Objective-C

Nel linguaggio Objective-C, gli oggetti sono identificati con un particolare tipo di dati definito *id*. Questo tipo è definito come un puntatore ad un oggetto, o meglio, come un puntatore ai dati dell'oggetto, alle sue variabili istanza. Ogni oggetto viene ad essere identificato attraverso il suo indirizzo e ad essere definito, indipendentemente dalle proprie variabili istanza o metodi, attraverso il tipo, completamente non restrittivo, id.

```
id unOggetto;
```

La sintassi riportata evidenzia come il puntatore *id* si limiti a definire l'oggetto *unOggetto* senza fornire informazioni ulteriori sui propri metodi o sulle proprie variabili istanza.

Il puntatore id, inoltre, svolge un ruolo fondamentale, assieme ai messaggi dell'oggetto, nel rendere disponibile all'Objective-C il requisito tipico della programmazione ad oggetti definito in precedenza come polimorfismo.

L'oggetto così definito non risulta unico in una programma ad oggetti: quest'ultimo sarà composto e basato su di una molteplicità di oggetti definibili attraverso la definizione delle loro classi. Questa operazione identifica un prototipo per ogni tipo di oggetto, dichiarando le variabili istanza che diverranno parte di ogni membro della classe e identificando l'insieme dei metodi che tutti gli oggetti della classe potranno usare.

Il compilatore crea solo un oggetto accessibile per ogni classe, un *class object* che conosce come costruire nuovi oggetti appartenenti alla classe. La class object rappresenta la versione compilata della classe e gli oggetti costruiti da questa costituiscono le *istanze* della classe, "istanziate" nella fase di esecuzione del programma.

Tutte le istanze di una classe hanno accesso allo stesso insieme di metodi condivisi dagli oggetti della classe mentre ogni oggetto avrà le proprie variabili istanza.

Per convenzione, i nomi delle classi iniziano con una lettera maiuscola mentre quelli delle istanze invece con una minuscola.

La definizione delle classi permette all'Objective-C di godere dei vantaggi dell'ereditarietà che trova riscontro nella possibilità di basare ogni nuova classe su di un'altra, da cui derivare metodi e variabili istanza. A tal riguardo l'Objective-C dispone di numerose classi potenzialmente utilizzabili dal programmatore come predefinite e pronte ad essere inserite nella propria applicazione oppure come basi da cui sviluppare opportune sottoclassi. Ciò consente l'elevato tasso di riutilizzabilità del codice tipico della programmazione ad oggetti.

L'albero di ereditarietà vede in cima alla scala gerarchica la classe *root* chiamata *NSObject* che definisce la struttura base degli oggetti, garantendo a questi le proprietà e caratteristiche in grado di qualificarli come tali all'interno dell'applicazione in esecuzione. Con la definizione di una classe si effettua un collegamento con la struttura gerarchica attraverso la dichiarazione della sua superclasse. Ogni classe che può essere definita deve essere una sottoclasse di un'altra, senza che sia stata definita una nuova classe root. Ogni nuova classe definita nell'applicazione può naturalmente utilizzare il codice scritto per tutte quelle che la precedono nella scala gerarchica.

In alternativa all'uso del tipo id, un oggetto può essere definito attraverso l'uso del nome della classe da cui deriva. Una simile dichiarazione viene definita come "tipizzazione statica" e fornisce al compilatore maggiori informazioni sull'oggetto definito attraverso la seguente sintassi:

```
Rettangolo *mioRettangolo;
```

In questo modo un oggetto è definito come un puntatore ad una classe, consentendo al compilatore di sottoporne a maggiori controlli il processo di creazione.

Definita la classe sarà possibile crearne nuove istanze attraverso un apposito metodo, il metodo *alloc*, che si occupa di gestire in maniera dinamica gli spazi di memoria per variabili istanza, inizializzate al valore zero, dell'oggetto creato.

Per creare una nuova istanza della classe Rettangolo assegnandola alla variabile mioRettangolo occorrerà utilizzare la seguente sintassi:

```
id mioRettangolo;
mioRettangolo = [Rettangolo alloc];
```

Una valida alternativa per inizializzare un oggetto utilizzando un maggior grado di specificità e personalizzazione è rappresentato dalla sintassi che utilizza il metodo *init* di solito collocato immediatamente dopo l'allocazione della memoria:

```
mioRettangolo = [ [Rettangolo alloc] init];
```

Questa istruzione utilizza contemporaneamente i due metodi: il primo (alloc) serve per creare il nuovo oggetto che, grazie al secondo (init), viene inzializzato. Una simile sintassi utilizza la classe sia per definire in maniera statica un tipo di oggetto e dall'altro come destinatario di un messaggio.

Nella definizione delle classi occorre particolare cautela dal momento che la loro visibilità globale potrebbe creare pericolose omonimie con le variabili globali. In questa fase è anche possibile per il programmatore ideare metodi non solo pensati per le istanze, ma anche per gli stessi oggetti classe, chiamati appunto "metodi di classe". In perfetta analogia con quelli generali anche quest'ultima tipologia di metodi gode delle possibilità concesse dall'ereditarietà di essere derivata dalle classi gerarchicamente dominanti.

In questa fase di ideazione delle nuova classe di oggetti sarà possibile definire quali variabili istanza dovrà avere. Ogni istanza della classe ha la sua copia delle variabili che vengono dichiarate ed ogni oggetto controlla i propri dati. Ciò comporta l'impossibilità di definire variabili classe da contrapporre alle variabili istanza dal momento che solo le strutture interne di dati, inizializzate dalla definizione della classe, sono disponibili per la classe. La classe oggetto, inoltre, non ha accesso diretto alle variabili istanza di alcuna istanza, essendo impossibilitata non solo ad inizializzarle, ma anche a leggerle o modificarle.

Questo limite implica l'adozione di variabili esterne per la condivisione dei dati tra le istanze; nel caso delle classi ciò avviene dichiarando variabili statiche e predisponendo i metodi necessari al loro utilizzo. Le variabili così definite non sono ereditate dalle sottoclassi e conferiscono agli oggetti classe un maggior livello di funzionalità rispetto alla semplice creazione delle istanze.

Se viene utilizzata una classe oggetto, è necessario provvedere alla sua inizializzazione in maniera analoga a quanto avviene per una istanza. Nonostante le applicazioni non prevedano l'allocazione della memoria per questo tipo di classi, l'Objective-C consente ai programmi di inizializzarle. Questi durante la fase di esecuzione inviano un particolare messaggio (*initialize*) a ogni classe oggetto prima che la classe riceva ogni altro messaggio. Ciò consente alla classe di predisporre il proprio ambiente di esecuzione prima che debba essere usato. Se l'inizializzazione non è necessaria, il metodo *initialize* non verrà utilizzato. Una particolarità riguarda il caso in cui la classe utilizzi variabili statiche o globali che potranno essere agevolmente inizializzate proprio attraverso questo costrutto.

Una volta che gli oggetti siano stati definiti, per poter dire loro di compiere un qualunque compito sarà necessario inviare un apposito *messaggio*

che contengo l'indicazione del metodo da applicare. La sintassi dell'Objective -C prevede che le espressioni che contengono messaggi siano definite, tra parentesi quadre, nel modo seguente:

```
[destinatario messaggio]
```

Il destinatario è un oggetto mentre il messaggio non è niente altro che quanto che si dice ad esso di compiere. Una volta che un messaggio sia stato inviato sarà il sistema che in fase di esecuzione provvede a selezionare i metodi appropriati tra quelli del destinatario e ad invocarli, eseguendoli.

Riprendendo l'esempio dell'oggetto definito in precedenza, un eventuale messaggio, ad esempio di visualizzarsi, passato a mioRettangolo potrebbe essere costruito nel modo seguente:

```
[mioRettangolo visualizzati];
```

Non è infrequente l'ipotesi di metodi che prevedano l'indicazione di alcuni argomenti come nel caso seguente in cui l'oggetto mioRettangolo riceve un metodo -*setbase:altezza:*- caratterizzato da due parametri con i quali si definiscono base e altezza:

```
[mioRettangolo setbase:  15.0 altezza:  20];
```

In linea teorica è possibile definire metodi che presentino un numero variabile di argomenti, separati da virgola al termine del nome del metodo, ma questa rappresenta una eventualità piuttosto remota.

Come le funzioni in C, i metodi dell'Objective-C possono restituire dei valori e non è infrequente che un messaggio sia nidificato all'interno di un altro come nel caso seguente in cui il colore del solito rettangolo viene definito con riferimento a quello di un altro:

```
[mioRettangolo setColorePrimario:

 [altroRettangolo colorePrimario]]
```

Un metodo ha accesso automatico alle variabili istanza dell'oggetto destinatario senza alcuna necessità di doverle passare al metodo stesso come argomenti.

Come illustrano gli esempi precedenti, i messaggi in Ojective-C sono posizionati nelle identiche posizioni a livello di sintassi delle chiamate alle funzioni nel C standard. Dal momento che i metodi appartengono sostanzialmente agli oggetti, è evidente che i messaggi hanno comportamenti differenti rispetto alla chiamata di un funzione. In particolare possiamo dire che un oggetto ha accesso esclusivamente ai metodi che gli sono stati definiti e ciò implica che due oggetti che abbiano due metodo diversi con lo stesso nome rispondano in maniera diversa allo stesso messaggio. Un simile comportamento appare sostanzialmente in linea con il polimorfismo tipico dei linguaggi di programmazione ad oggetti.

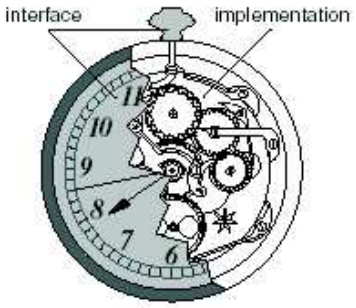

Figura 3.1: Interfaccia ed implementazione

3.3.2 Interfaccia e implementazione di una classe

Buona parte della programmazione ad oggetti consiste nello scrivere il codice per nuovi oggetti e definire nuove classi; queste operazioni in Ojective-C prevedono la definizione di:

- un'*interfaccia* che dichiara i metodi e le variabili di istanza della classe con precisa indicazione dei nomi della superclasse;
- un'*implementazione* che definisce concretamente la classe contenendo il codice che rende operativi i propri metodi.

Nonostante non sia richiesto dal compilatore, generalmente interfaccia e implementazione sono definiti in due differenti file, uno solo dei quali, quello dell'interfaccia, deve essere reso disponibile a chiunque usi la classe. Quello dell'implementazione non dispone dello stesso grado di diffusione dal momento che l'utilizzatore non necessita del relativo codice sorgente durante questa fase.

Sebbene sia possibile dichiarare più di una classe all'interno dello stesso file, per rendere più agevole la comprensione codice si tende a separare i file di interfaccia e di implementazione di ogni classe. Questa scelta riflette la volontà di porne in evidenza lo status di entità indipendenti, seguendo i canoni della programmazione ad oggetti.

In linea con queste finalità, file di interfaccia e di implementazione risultano caratterizzati da differenti estensioni: il primo ha suffisso ".h" (tipico degli *header file*), mentre il secondo ".m" che ne indica al compilatore il contenuto in termini di codice sorgente.

Riprendendo l'esempio precedente, la classe Rettangolo potrebbe essere dichiarata in un file Rettangolo.h e definita in Rettangolo.m.

Il file di interfaccia, il quale contiene le informazioni che qualificano la classe, permette a quest'ultima di essere dichiarata agli altri "moduli" che costituiscono il programma. Nei file che hanno estensione .h infatti si trovano

tutte le informazioni necessarie per identificare la classe all'interno della catena di ereditarietà, per definire quali variabili istanza siano contenute nell'oggetto e per conoscere quali messaggi potrà ricevere.

L'oggetto così definito rappresenta una entità auto-organizzata che può essere vista dall'esterno come una sorta di "scatola nera". Una volta determinato come un oggetto interagirà con gli altri elementi del programma, ossia una volta che sarà definita la sua interfaccia, sarà possibile modificarne la implementazione senza dover rivedere le altre parti dell'applicazione.

Dal momento che tutte le istruzioni che debbono essere inviate al compilatore in Objective-C iniziano con "@", la dichiarazione dell'interfaccia di una classe inizia con la sintassi `@interface` e termina con `@end`.

```
@interface NomeClasse : SuaSuperClasse

{

dichiarazione delle variabili istanza

}

dichiarazione dei metodi

@end
```

La prima riga della dichiarazione definisce il nome della nuova classe e la collega alla superclasse da cui deriva, definendone la posizione nella gerarchia dell'ereditarietà. Qualora non sia indicata la superclasse allora la nuova classe che viene definita è dichiarata come una classe *root*. Con la dichiarazione delle variabili istanza, racchiuse tra parentesi graffe, viene definita la struttura dei dati che avrà ogni istanza della classe. Successivamente si passa alla dichiarazione dei metodi i cui nomi nel caso siano usati dagli oggetti della classe (*metodi di classe*) saranno preceduti dal segno "+" mentre nel caso di utilizzo da parte delle istanza della classe (*metodi di istanza*) quello "-". In linea teorica potrebbe essere possibile assegnare a due metodi dei due tipi uno stesso nome, ma motivi pratici e formali rendono questa pratica di scarsa applicazione. Per la definizione dei "tipi" di dati restituiti dal metodo si ricorre alle dichiarazioni tipiche della sintassi del C standard, con la implicita assegnazione al tipo `id` in caso di non espressa dichiarazione.

Il file dell'interfaccia deve essere incluso in ogni modulo di codice che dipende dall'interfaccia della classe. Questa inclusione viene fatta con il comando `#import` (molto simile a `#include`) con la seguente sintassi: `import ‘‘SuaSuperClasse.h’’` Per riflettere la definizione di una classe è costru-

ita sulle dichiarazioni delle classi ereditate, un file di interfaccia inizia con l'importazione dell'interfaccia della sua superclasse:

```
#import ‘‘SuaSuperClasse.h’’

@interface NomeClasse: SuaSuperClasse

{

dichiarazione delle variabili istanza

}

dichiarazione dei metodi

@end
```

Questa sintassi implica che ogni file di interfaccia includa, in maniera derivata, tutte le interfacce delle classi da cui eredita, seguendo la intera gerarchia dell'ereditarietà. Quando una interfaccia fa riferimento a classi che non sono nella sua gerarchia, è necessario importarle in maniera esplicita oppure dichiararle facendo ricorso al comando `@class`:

```
@class Rettangolo, Cerchio;
```

In questo modo si informa il compilatore che "Rettangolo" e "Cerchio" sono due nomi di classi senza importarne i relativi file di interfaccia. Questi ultimi potranno poi essere importati attraverso l' implementazione dell'oggetto appena definito.

Il ruolo del file di interfaccia è quello di dichiarare la nuova classe agli altri componenti del codice che compone l'applicazione, contenendo tutte le informazioni che questi ultimi necessitano per lavorare con la classe.

Le tre principali funzioni del file di interfaccia possono riassumersi cosi:

- informare l'utilizzatore su come la classe è connessa all'interno della gerarchia dell'ereditarietà e su quali classi -ereditate o solamente citate all'interno della classe- siano necessarie;
- permettere al compilatore di conoscere quali variabili istanza siano contenute nell'oggetto e al programmatore di sapere quali variabili la loro sottoclasse erediterà;
- permettere agli altri componenti del programma, attraverso la propria lista di dichiarazione dei metodi, di essere informati su quali messaggi possono essere mandati ad un oggetto classe e alle istanze della classe.

Dichiarata l'interfaccia della classe, occorre passare alla definizione del file dell'implementazione che inizia, in analogia con quanto visto in precedenza, con l'istruzione `@implemtation` e termina con `@end`.

```
@implementation NomeClasse : PropriaSuperClasse

{

dichiarazione delle variabili istanza

}

definizione dei metodi

@end
```

Ogni file di implementazione deve importare la propria interfaccia ma può tralasciare di riportare il nome della superclasse e le dichiarazioni delle variabili istanza. In questo modo la definizione dell'implementazione risulta semplificata e maggiormente volta alla definizione dei metodi:

```
#import‘‘NomeClasse.h’’

@implementation NomeClasse

definizione dei metodi

@end
```

I metodi di una classe, come le funzioni in C, sono racchiusi all'interno delle parentesi graffe, dopo essere stati opportunamente dichiarati nello stesso modo in cui avviene nel file di interfaccia:

```
+ alloc

{

...

}

- (BOOL) isfilled

{
```

```
}

- (void) setFilled: (BOOL) flag

{

...

}
```

Secondo l'impostazione generale, la definizione di un metodo istanza rende disponibili tutte le variabili istanza dell'oggetto, rendendo possibile l'accesso a queste semplicemente per nome senza necessità della dichiarazione. Quando però si renda necessario utilizzare una variabile istanza cha appartiene ad un oggetto che non è il ricevente del messaggio, il tipo dell'oggetto deve essere reso esplicito al compilatore con la notazione della tipizzazione statica. Per riferirsi ad una variabile istanza di un oggetto tipizzato staticamente occorrerà far ricorso all'operatore di puntatore alla struttura ('$->$').

Nonostante la loro dichiarazione avvenga a livello dell'interfaccia della classe, le variabili istanza sono più legate al modo in cui la classe è implementata che a quello in cui è utilizzata. Nell'interfaccia, infatti, si trova l'elencazione dei metodi, ma non la definizione della struttura interna dei dati.

Con la finalità di accrescere la possibilità di un oggetto di nascondere i propri dati, il compilatore limita l'ambito di operatività delle variabili istanza, controllando la loro visibilità all'interno del programma. Tuttavia, la flessibilità dell'Objective-C permette al programmatore di definire l'ambito su tre possibili livelli, attraverso specifici comandi del compilatore:

- **@private**: la variabile istanza è accessibile solamente all'interno della classe che la dichiara;
- **@protected**: la variabile istanza è accessibile all'interno della classe che la dichiara e delle classi che ereditano da questa;
- **@public**: la variabile istanza è accessibile ovunque.

In linea generale, le variabili istanza sono di tipo @protected se non espressamente definite.

Le motivazioni per cui si possono voler introdurre restrizioni alle classi ereditate nell'accesso alle variabili istanza possono essere così sintetizzate:

- una volta che la sottoclasse accede ad una variabile istanza, la classe che dichiara la variabile è legata a quella componente della sua implementazione; in una versione del programma successiva, non si potrà eliminare la variabile o modificarne la funzione senza inavvertitamente cambiare la sottoclasse;

- se una sottoclasse accede alla variabile istanza ereditata e ne altera il valore, possono essere introdotti inavvertitamente errori all'interno della classe che dichiara la variabile, specialmente se la variabile è coinvolta da dipendenze all'interno della classe.

Per limitare l'ambito di una variabile istanza alla sola classe che la dichiara, sarà sufficiente contrassegnarla come @private mentre all'altro estremo identificandola come @public la si renderà disponibile all'esterno delle definizioni di classe che la ereditano o la dichiarano. In quest'ultimo caso gli altri oggetti potranno ottenere le informazioni contenute in una variabile istanza semplicemente con un messaggio di richiesta. Rendere @public una variabile fa venir meno la possibilità per l'oggetto di rendere nascosti i propri dati: questa conseguenza nega uno dei cardini della programmazione ad oggetti -l'incapsulazione- e dovrebbe essere evitata tranne in casi eccezionali.

Nel linguaggio Objective-C, i messaggi non sono fatti avanzare all'implementazione dei metodi fino alla fase di esecuzione dell'applicazione. Il compilatore converte un espressione di messaggio come la seguente,

`[destinatario messaggio]`

in una chiamata a una funzione di messaging, $objc_msgSend()$.

Questa funzione, che ha come argomenti il destinatario e il nome del metodo citato nel messaggio (*method selector*), esegue le operazioni necessarie per il dynamic binding:

1. trova la procedura (implementazione del metodo) a cui si riferisce il selector;

2. chiama la procedura, passandole l'oggetto ricevente (un puntatore ai suoi dati), con gli argomenti che sono stati specificati per il metodo;

3. passa il valore di ritorno della procedura come il proprio valore da restituire.

Il punto chiave del processo di messaging si ritrova nelle strutture, un puntatore alla superclasse ed una apposita classe detta *dispatch table*, che il compilatore costruisce per ogni classe e per ogni oggetto.

Quando un nuovo metodo viene creato, viene predisposta una zona di memoria per allocarlo e vengono inizializzate le variabili, prima tra tutte uno speciale puntatore, chiamato *isa*, alla sua struttura di classe.

Quando un messaggio viene mandato ad un oggetto, la funzione di messaging indirizza il puntatore isa dell'oggetto alla sua struttura di classe dove

interroga il selector dei metodi nella dispatch table. Una volta trovato il selector, chiama il metodo ottenuto dalla dispatch table e gli passa la struttura dei dati dell'oggetto ricevente.

L'Objective-C, inoltre, dispone di due comandi che possono essere usati all'interno della definizione dei metodi per far riferimento all'oggetto che deve eseguire il comando: **self** e **super**. Queste istruzioni, contenute nei messaggi inviati ai metodi degli oggetti, indicano al compilatore in quale classe occorre cercare l'implementazione del metodo da eseguire.

Self è uno degli argomenti nascosti che la routine di messaging passa ad ogni metodo e può essere sostituito da super solo come destinatario di un'espressione di messaggio. Dal punto di vista funzionale i due comandi differiscono sostanzialmente sulla base delle modalità in cui eseguono il processo di messaging:

- *self*: cerca l'implementazione del metodo indicato nel messaggio nella maniera standard ossia partendo dalla dispatch table dalla classe dell'oggetto ricevente;

- *super*: inizia la ricerca dell'implementazione del metodo iniziando dalla superclasse della classe che definisce il metodo dove è contenuto super.

Per fare un esempio, se si dispone di una struttura gerarchica di tre livelli con una classe e una superclasse dotate di un metodo con un identico nome *negoziate* e se tale metodo viene richiamato all'interno di un altro metodo *makeLastingPeace*, definito nella sottoclasse, possono aversi le due opzioni. Se l'implementazione del metodo *makeLastingPeace* è:

```
- makeLastingPeace  [self negoziate]
```

l'applicazione cerca il metodo *negoziate* nella sottoclasse.

Se, invece, l'implementazione è:

```
- makeLastingPeace  [super negoziate]
```

in questo caso il metodo viene ricercato nella superclasse e l'istruzione super fa si che non si tenga conto del metodo definito nella sottoclasse.

3.4 Confronto tra Objective-C e Java

Una evoluzione del progetto Swarm è stata rappresentata dall'averne riscritto le librerie, originariamente basate sul linguaggio Objective-C, in Java. Nonostante i due linguaggi, entrambi estensioni ad oggetti della sintassi del C standard, presentino notevoli affinità, è possibile rilevare alcune differenze sostanziali nella scrittura di applicazioni Swarm:

- l'Objective-C ha conservato, a differenza di Java, tutte le principali caratteristiche a basso livello del C e che sono utilizzate in primis per lo sviluppo dei sistemi operativi;

- in Java, variabili e tipi devono sempre essere specificati (tipizzazione statica). Se per certi versi questo rappresenta un limite alla flessibilità della programmazione da un lato, dall'altro lato aumenta le forme di controllo e di prevenzione degli errori in fase di compilazione. In Ojective-C, dal momento che molte attribuzioni avvengono nella fase di esecuzione, può capitare frequentemente che l'applicazione provochi un errore nella fase del runtime dovuto ad una errata assegnazione. Ciò non accade in Java dove il compilatore ha il compito di effettuare un controllo preventivo sulla coerenza delle varie assegnazioni;

- una applicazione risulta più lenta se scritta in Java che in Objective-C: questo è dovuto al fatto che il codice Java viene prima tradotto in fase di compilazione in un metacode (*bytecode*), compatibile con ogni piattaforma, e successivamente viene specifico dalla specifica piattaforma utilizzata come sistema operativo;

- i due linguaggi presentano una diversa modalità di gestione e allocazione delle zone di memoria. Java dispone di una utilissima funzione denominata *garbage collection*, capace di eliminare gli oggetti non più utilizzati, alleggerendo l'esecuzione dell'applicazione. In Objective-C tale funzionalità risulta assente e l'eliminazione degli oggetti non più in uso non può avvenire in maniera automatica, ma l'operazione deve essere compiuta con opportune istruzioni scritte nel codice dell'applicazione. Questo implica un attento controllo degli oggetti non più utilizzati e delle relative zone di memoria durante la fase di esecuzione da parte dello sviluppatore: spetterà a quest'ultimo il compito di eliminare quelli non più necessari, liberando risorse dell'elaboratore con il duplice effetto di accelerare l'esecuzione ed evitare pericolosi blocchi o instabilità del sistema;

- la compatibilità con le più diffuse piattaforme costituisce un indubbio vantaggio dell'utilizzo di Java;

- nonostante il forte sviluppo che la comunità Swarm impegnata a scrivere codice in Java, molte applicazioni e librerie sono disponibili esclusivamente in Objective-C, il linguaggio nativo del progetto;

- come diretta conseguenza del punto precedente, la comunità degli sviluppatori di Swarm ha dedicato notevoli risorse all'apprendimento della programmazione in Objective-C, acquisendo competenze e creando modelli ormai consolidati; tuttavia, quest'ultimo linguaggio non ha avuto una vasta diffusione tra i programmatori a differenza di Java. A determinare il forte sviluppo di Java è stata la sua caratterizzazione di linguaggio internet-oriented.

Le differenze concettuali che caratterizzano i principi di programmazione dei due linguaggi determinano notevoli differenze anche da un punto di vista formale nella definizione dell'applicazione e nella scrittura del relativo codice.

3.5 Swarm

Identificati canoni e caratteristiche della programmazione ad oggetti facendo costante riferimento all'Objective-C, non resta che approfondire la conoscenza di Swarm per essere in grado di creare simulazioni ad agenti con questo flessibile e potente strumento di sviluppo.

Sin dalla fase di definizione del progetto, gli ideatori di Swarm hanno voluto mettere a disposizione della comunità scientifica un ambito nel quale far agire e soprattutto interagire un elevato numero di soggetti, o meglio "agenti". Questa volontà ha trovato riflesso nelle scelte informatiche compiute per la definizione del pacchetto di librerie software che ne è risultato.

Swarm, in linea con il paradigma della simulazione basata su agenti, è stato ideato per la creazione di oggetti ordinati gerarchicamente. L'*observer Swarm* è il primo ad essere creato e da questo si genera una *user interface* e viene "istanziato" il *model Swarm*. L'istanza del model Swarm, a sua volta, creerà gli oggetti di livello inferiore e si occuperà di organizzare le loro attività, gestendone lo *schedule*.

In questa attività di definizione del modello Swarm, il programmatore è favorito dalla capacità dello strumento di creare codice di alta qualità con un limitato livello di difficoltà: la libreria di Swarm crea una sequenza di classi che accrescono e migliorano la capacità di ideare oggetti di simulazione, di gestirne la memoria e di organizzarne e programmarne le attività.

Un aspetto particolarmente interessante di Swarm è rappresentato dalla sua capacità di gestire in maniera semplice, ma efficace il tempo della simulazione. Utilizzando un linguaggio di programmazione standard per scrivere la propria simulazione, il programmatore dovrà dedicare molte risorse alla gestione della memoria e soprattutto alla definizione di opportune iterazioni in grado di regolare il tempo degli eventi. Questi problemi possono essere evitati utilizzando Swarm che si fa carico di queste attività e, trattando eventi ed agenti allo stesso modo[2], libera risorse dello sviluppatore da destinare alla definizione del modello.

L'elemento essenziale di una simulazione Swarm è costituito dall'agente: all'interno del modello i singoli agenti, definiti con interfaccia e implementazione diverse tra loro, vengono fatti interagire realizzando sistemi complessi.

In analogia con quanto avviene nei linguaggi di programmazione ad

[2]Sia eventi che agenti vengono definiti come oggetti e questo risulta particolarmente utile i primi la cui realizzazione può essere programmata grazie ad opportuni strumenti sensibili al trascorrere del tempo.

oggetti, la definizione del singolo agente avviene a livello di classe. Dalla classe, grazie alle proprietà di ereditarietà, incapsulamento e polimorfimo, potrà essere definita una molteplicità di istanze, i singoli agenti, che costituiranno la popolazione della simulazione.

L'organizzazione e la gestione degli agenti è demandata ad un oggetto fondamentale, definito appunti *Swarm*, che contiene al suo interno sia gli agenti che la rappresentazione del tempo. Nello Swarm è definito un insieme di agenti ognuno dotato del proprio registro (*schedule*) di funzioni da compiere.

La particolare modalità di gestione del tempo che caratterizza Swarm fa sì che le azioni dei singoli agenti che compongono la simulazione non siano collocati in un tempo astratto, ma abbiano luogo in un tempo definito. La scansione degli eventi, regolata dalle strutture dei dati che determinano lo schedule, definisce il trascorrere del tempo. Nello schedule, infatti, sono raggruppate le azioni dei singoli agenti nell'apposito ordine di esecuzione; questi da un punto di vista informatico coincidono con i metodi che devono essere applicati sui vari oggetti destinatari.

Allargando l'ottica di osservazione, lo stesso Swarm, appena definito, può essere definito come un agente il cui comportamento emergerà dalle azioni interconnesse dei singoli agenti in esso contenuti. In questo modo sarà possibile definire modelli gerarchici semplicemente annidando una molteplicità di Swarm. Questo aspetto conferisce a Swarm un elevato grado di flessibilità, estendendo le potenzialità messe a disposizione dallo strumento per gli sviluppatori.

3.5.1 Struttura di una simulazione realizzata con Swarm

Lo sviluppatore di modelli Swarm, nel proprio lavoro, segue in linea generale uno schema come il seguente:

1. crea un ambiente artificiale definito in termini di spazio-tempo e dotato di oggetti allocati nella struttura che sono in grado di definire il proprio stato secondo specifiche regole di comportamento;

2. definisce strumenti in grado di osservare, registrare ed eventualmente analizzare le strutture di dati che emergono dai comportamenti degli agenti, oggetti del mondo simulato;

3. esegue la simulazione;

4. interagisce con l'esperimento attraverso ad opportuni oggetti definiti *"probe"* che consentono controlli sperimentali, analizzando le strutture interne degli oggetti che compongono la simulazione.

Determinante ai fini della riuscita dell'esperimento risulta quindi il successo nella definizione del modello con l'identificazione della sua struttura e

dei suoi componenti. Il grado di complessità del modello, inizialmente definito in maniera semplice con un unico Swarm dotato di un solo gruppo di agenti ed un solo schedule di attività da compiere, può essere successivamente incrementato con estrema facilità.

La notevole flessibilità di Swarm risulta ancor più apprezzabile nella fase di definizione del modello e nella fase di modifica grazie alla possibilità di definire l'ambiente artificiale senza dover utilizzare strutture già definite ed immutabili. L'ambiente, come qualunque altro oggetto della simulazione, può essere definito e modificato per tener conto delle caratteristiche volute dall'ideatore del modello, senza dover rivedere le altre componenti dell'applicazione.

Una volta definiti gli oggetti della simulazione (agenti e struttura del modello), questi costituiscono un oggetto Swarm quando sia definito lo schedule della attività che definisce il tempo della simulazione.

Creato l'oggetto Swarm, per poter analizzare dati e comportamenti della realtà simulata è necessario definire una serie di strumenti a cui è affidata questa importantissima attività:

- un observerSwarm, oggetto anch'esso dotato di un proprio schedule, che osserva il comportamento della simulazione, registrando i dati delle varie attività;
- un insieme di probe (o sonde), con cui controllare lo stato degli oggetti durante l'esecuzione della simulazione.

Risulta evidente come il tratto caratterizzante di una simulazione impostata secondo questi canoni sia una struttura che combina l'observerSwarm, e di riflesso l'osservatore, con il modello ideato. Le due fasi, sperimentazione e osservazione, formalmente definite da due oggetti separati, rispettivamente il modelSwarm e l'observerSwarm, grazie alla definizione di relativi schedule[3] riescono ad essere sincronizzate. Questa efficace gestione del timing della simulazione rappresenta un tratto peculiare e vincente di Swarm nei confronti dei linguaggi di programmazione tradizionali nell'ambito dei modelli basati su agenti.

3.5.2 Nozioni fondamentali sulla sintassi Swarm in Objective-C

I modelli Swarm utilizzano una semplice sintassi che aiuta gli utilizzatori a capire le modalità con cui interagiscono le singole parti. Una prima semplificazione è dovuta al fatto che observer e modelSwarm sono concepiti in

[3]Gli schedule di observerSwarm e modelSwarm vengono realizzati all'interno dei metodi *buildActions* presentano una struttura sostanzialmente simile. Nello schedule del modelSwarm sono presenti i metodi *modelActions* e *modelSchedule* mentre nell'observerSwarm si trovano gli omologhi *displayActions* e *diplaySchedule*. A caratterizzarli sono i messaggi passati ed i relativi destinatari.

maniera molto simile. Per fare un esempio, i metodi che possono apparire in molte classi includono le seguenti istruzioni:

```
-createBegin; -createEnd; -buildObjects; -buildActions;
-activateIn;
```

Sono inoltre presenti metodi che consentono l'invio e la ricezione di informazioni dal singolo oggetto: convenzionalmente, questi metodi sono rispettivamente preceduti dai prefissi "get" e "set".

```
-setValoreParametro: (int) valore; -(int) getValoreParametro;
```

Il modelSwarm è di solito una sottoclasse che eredita dalla classe Swarm ed è l'oggetto principale che si occupa di gestire il processo di creazione dei singoli agenti da parte delle varie sottoclassi. Il model, inoltre, si fa carico di assegnare agli agenti creati una apposita zona di memoria e di organizzarne le attività pianificate.

Le attività di gestione della memoria rappresentano una componente fondamentale e molto spesso critica di ogni simulazione: fortunatamente Swarm mette a disposizione degli sviluppatori una serie di librerie che rendono il processo estremamente trasparente e allo stesso tempo flessibile.

Osservando un applicazione in Swarm, risulta evidente l'importanza degli aspetti legati alla creazione degli oggetti, quando un oggetto viene definito e viene allocato in una apposita zona di memoria. Per svolgere questa importante attività, Swarm dispone del metodo `buildObjects` che non si limita a creare l'oggetto usato nella classe corrente, ma rappresenta l'istruzione che controlla il processo di creazione degli oggetti da parte degli agenti del livello inferiore più prossimo.

Analizzando la costruzione delle azioni, non è possibile prescindere dal metodo `buildActions` che gode della stessa importanza di quello appena citato dal momento che crea oggetti delle due classi fornite dalla libreria *activity*:

- ActionGroup: un insieme ordinato di eventi simultanei[4]; costituisce una lista di eventi che associa ad ogni oggetto destinatario il metodo da eseguire;
- Schedule: che controlla la frequenza di esecuzione degli elementi presenti in ActionGroup; rappresenta una struttura di dati che elenca le azioni degli oggetti in uno specifico ordine di esecuzione.

Le due classi così definite vengono rese operative dal metodo `buildActions` che opera nel modo seguente:

- crea uno o più gruppi di azioni grazie ad ActionGroup;

[4]Poiché Swarm non realizza una elaborazione parallela dei processi in esecuzione, gli eventi di *actionGroup* sono eseguiti in maniera sequenziale.

- crea lo schedule che definisce la scansione del tempo e il *time-step*, l'intervallo di ripetizione degli eventi[5].

Il metodo `buildActions` può essere presente in più classi e dal momento che è possibile creare molteplici ActionGroup e schedule, è necessario mantenere la coerenza di tutte le loro attività. Ciò viene fatto a livello dello schedule principale dal momento che ogni schedule definito a livello di sottoclasse di Swarm viene sincronizzato con quello del livello gerarchico più elevato da cui deriva. Risalendo in questo modo la scala della gerarchia, ogni schedule risulta sincronizzato con quello del livello Swarm, il più elevato.

3.5.3 Descrizione della struttura di un programma in Swarm

Elementi caratterizzanti di una applicazione Swarm sono:

- il file *main*;
- l'*observerSwarm*;
- il *modelSwarm*;
- i file che definiscono gli agenti del modello;
- il Makefile.

Il *main* è il file più importante dell'applicazione che ha lo scopo di inizializzare lo Swarm e di lanciare la simulazione. La sequenza delle attività svolte generalmente da questo componente può essere così sinterizzata:

1. creazione dell'observerSwarm e allocazione in una apposita zona di memoria;
2. creazione degli oggetti dell'observer con il metodo buildObjects;
3. creazione delle azioni dell'observer con il metodo buildActions;
4. attivazione dell'observer;
5. avvio della simulazione.

La sequenza della attività trova riflesso nel listato del main qui riportato:

```
#import <simtools.h>

#import ‘‘ObserverSwarm.h’’
```

[5]L'intervallo di ripetizione viene a coincidere con il numero di time-step tra la chiamata della prima azione di un processo e l'inizio di uno nuovo.

```
int main (int argc, const char **argv)

{

ObserverSwarm *observerSwarm;

initSwarmApp (argc, argv);

observerSwarm = [ObserverSwarm create: globalZone];

[observerSwarm buildObjects];

[observerSwarm buildActions];

[observerSwarm activateIn: nil];

[observerSwarm go];

return 0;

}
```

Nel main non è presente alcun riferimento al modelSwarm, che invece viene creato all'interno dell'observerSwarm assieme ad altri oggetti necessari per l'osservazione della simulazione:

- le probe dell'observer;
- il modelSwarm;
- i grafici e le sonde;
- il pannello di controllo, che consente all'utilizzatore di agire sul funzionamento della simulazione, avviandola, arrestandola o salvandone i risultati.

Da un punto di vista del codice, l'observer risulta così caratterizzato:

```
// ObserverSwarm.h

#import ‘‘ModelSwarm.h’’;

#import <simtoolsgui/GUISwarm.h>

#import <analysis.h>
```

```
@interface ObserverSwarm: GUISwarm

{ int displayFrequency;

...

id displayActions;

id displaySchdule;

id <EZGraph> Graph1, Graph2;

ModelSwarm * modelSwarm

}

+ createBegin: aZone;

-createEnd;

-buildObjects;

-buildActions;

-activateIn: SwarmContext;

-checkToStop;

....;

@end
```

E' importante rilevare che tra tutti gli oggetti creati dall'observer solo la sua probe non viene definita all'interno del metodo *buildObjects*, ma in quello *createBegin*.

Lo schedule dell'obeserver, in perfetta coerenza con quanto visto in precedenza, viene definito all'interno del metodo buildActions che prima di questa operazione manda un analogo messaggio al modelSwarm per la creazione del proprio schedule.

La classe modelSwarm è quella che contiene la descrizione del modello con cui vengono creati gli oggetti necessari per la realizzazione della simulazione. Oltre ad uno schedule, già citato, per la definizione del tempo del

modello anche questa classe può disporre di una propria probe che definisce parametri e variabili della simulazione. Ecco un esempio di modelSwarm:

```
//ModelSwarm.h

#import <objectbase/Swarm.h>

#import <simtools.h>

#import <activity.h>

#import <collections.h>

@interface ModelSwarm: Swarm

{

int dayNumber, int agentNumber,...

float ...

id <ActionGroup> modelActions1;

id <Schedule> modelSchedule;

...

}

+ createBegin: aZone;

-createEnd;

-buildObjects;

-buildActions;

-activateIn: SwarmContext;

-increaseCurrentDayNumber;

-(int) getCurrentDay;
```

```
-...

-openProbeTo: (int) ag;

@end }
```

Accanto ai file che contengono model ed observer, una simulazione Swarm necessita del codice necessario a definire interfaccia e implementazione delle classi di agenti che vengono dichiarati. In analogia con quanto avviene per l'Objective-C i due aspetti dell'oggetto creato si trovano rispettivamente in file con estensione ".h" e ".m".

Definite le componenti fondamentali della simulazione, i processi di linking e di compilazione del codice si realizzano grazie alle istruzioni contenute in un apposito file denominato *Makefile*. Durante il processo di compilazione, avviato grazie al comando "make" lanciato al prompt di Swarm nella cartella del programma, il codice contenuto in interfaccia e implementazione viene utilizzato per creare i file *object* in formato binario e caratterizzati da estensione ".o". La successiva fase di linking, utilizza questi file, unendoli, per creare il file, con estensione ".exe", eseguibile dall'utilizzatore dell'applicazione.

3.6 Le librerie disponibili in Swarm

Grazie alla ampia disponibilità di librerie, i programmatori di modelli in Swarm hanno la facoltà di creare oggetti avvalendosi delle classi già realizzate e rese pubbliche dalla comunità degli sviluppatori e degli utilizzatori. In linea con i principi di riutilizzabilità del codice che caratterizzano la programmazione ad oggetti, per definire un oggetto è sufficiente crearne una istanza da una delle classi già sviluppate. Questa possibilità risulta estremamente utile per definire oggetti come schedule e interfaccia grafica, particolarmente onerosi di risorse necessarie alla realizzazione. Le librerie, ormai parte fondamentale del pacchetto software, fornite dagli stessi autori di Swarm sono qui di seguito illustrate in maniera sintetica.

- *defobj*: rappresenta la classe root nella gerarchia delle librerie SWARM, contenendo le funzionalità che permettono la definizione dei metodi di base per la creazione e l'eliminazione degli oggetti e le classi che consentono di archiviare le istanze degli oggetti;
- *objetcbase*: libreria che contiene le classi fondamentali su cui si basa il funzionamento degli agenti. Tre le varie classi presenti definite nella libreria occorre citare *SwarmObject*, la classe radice (*root*) da cui ereditano tutti gli agenti simulati, e *Swarm*, che fornisce le funzionalità richieste per la definizione di modelSwarm ed observerSwarm;

- *activity*: libreria già citata in precedenza, contiene il nucleo delle funzioni che gestiscono le strutture di dati necessarie a definire la sequenza degli eventi, permettendo la creazione dello schedule;
- *simtools*: fornisce le classi indispensabili per il controllo del processo simulativo;
- *probe*: contiene le classi necessarie alla definizione degli oggetti sonda, utili per osservare e registrare variabili e parametri degli agenti della simulazione;
- *collections*: mette a disposizione le classi necessarie per la creazione e la gestione di liste, vettori e mappe;
- *random*: contiene una utile raccolta di generatori di numeri casuali definiti secondo gli algoritmi matematici tratti dalla letteratura in materia;
- *gui*: fornisce le classi che permettono la creazione di oggetti grafici ed immagini e gestisce l'iterazione dell'utilizzatore con l'interfaccia grafica;
- *tkobjc*: la realizzazione e la gestione dell'interfaccia grafica è resa possibile dalle classi di questa libreria, basate su Tcl/Tk.

Oltre alle librerie espressamente fornite dagli sviluppatori di Swarm, la comunità degli utilizzatori ha apportato il proprio contributo attraverso numerose pacchetti software "aggiuntivi" che mettono a disposizione svariati algoritmi sviluppati per scopi specifici.

Capitolo 4

I modelli ABM e i mercati finanziari

4.1 Introduzione

Nell'ultimo decennio, sono stati sviluppate un gran numero di simulazioni di mercati finanziari simulati. Seguendo le orme del modello pionieristico ASM con il quale gli studiosi del Santa Fe Institute hanno mostrato l'applicabilità delle tecniche di modellizzazione ad agenti nello studio dei mercati finanziari, un numero sempre maggiore di ricercatori ha proposto modelli artificiali popolati da agenti eterogenei dotati di capacità di apprendimento e di ottimizzazione del proprio comportamento. In alcuni molti casi questi modelli hanno evidenziato comportamenti molto simili a quelli del mondo reale con buona capacità di riprodurne aspetti peculiari, ma si sono spesso mostrati complessi in termini di studio analitico.

Altri ricercatori hanno sviluppato mercati artificiali popolati sempre da agenti eterogenei, ma caratterizzati da regole di azione semplici. Queste simulazioni si sono dimostrate più facilmente modellizzabili e interpretabili da un punto di vista analitico, mantenendo intatte le capacità di rappresentare aspetti fondamentali dei comportamenti dei mercati. Allo stato attuale, si è ancora lontani dal poter dire che un mercato artificiale sia in grado di spiegare tutte le assunzioni disponibili in termini di comportamento delle attività finanziarie. Data la complessità del compito, i ricercatori impegnati in questi progetti sono chiamanti a scelte e compromessi tra semplici generalizzazioni e una fedele rappresentazione degli aspetti reali.

In questo capitolo, rifacendosi agli strumenti e alle impostazioni metodologiche espresse in quelli precedenti, si analizzano due modelli, l'ASM e il *Genoa Market* che, partendo da assunzioni diverse, forniscono due esempi di simulazioni applicate ai mercati azionari. Il terzo modello, *SUM*, sarà oggetto del capitolo successivo in virtù delle caratteristiche peculiari che lo distinguono dai due citati in precedenza. Anticipando sinteticamente l'in-

troduzione del terzo modello, è possibile affermare che il principale aspetto distintivo tra SUM, da un lato, e ASM e *Genoa Market*, dall'altro, deriva dal confronto dei meccanismi endogeni di formazione del prezzo delle attività scambiate: nel modello SUM la formazione del prezzo di scambio avviene transazione dopo transazione, sulla base del confronto delle proposte di segno opposto, immesse nel sistema dagli operatori, mentre negli altri due modelli il valore attribuito dal mercato alle attività è frutto del confronto di domanda e offerta aggregata espressa dagli agenti.

4.2 Il modello ASM, *Artificial Stock Market*

Risultato dell'opera di Arthur, Holland, LeBaron, Tayler e Palmer, il modello ASM, ha rappresentato ai tempi del propria sviluppo (1997) una delle prime simulazioni ad agenti[1] applicata ai mercati finanziari. Caratteristiche distintive del modello sono un forte rigore formale nel simulare la struttura del mercato e l'utilizzo delle tecniche dei *classifier system* per rappresentare i processi cognitivi degli agenti.

Sebbene l'evidenza empirica sembri confermare che gli operatori, nel processo di allocazione del risparmio, utilizzino e sottopongano a costante aggiornamento schemi di investimento semplici per ottimizzare i propri investimenti, mantenendoli adeguati alle variazioni di condizioni sui mercati, gli autori hanno ideato i loro agenti seguendo il paradigma delle aspettative razionali.

In linea con questa impostazione, gli operatori simulati effettuano le scelte di investimento tendendo conto della loro esperienza passata e sulla base delle aspettative sull'andamento del mercato. Le aspettative saranno frutto della capacità di previsione dei singoli agenti, capaci di effettuare un processo di scelta delle informazioni ritenute rilevanti nel processo.

Il modello di chiara derivazione neoclassica si distingue dai modelli tradizionali, tipicamente descrittivi, per l'introduzione della figura dell'agente capace di formulare aspettative dinamiche; questo sono capaci di adeguarsi al variare della condizioni del mercato sulla base di regole eterogenee, frutto del processo evolutivo degli schemi realizzato dal *classifier system*. La formazione di specifiche formulazioni previsionali che caratterizza ogni agente ha profondi riflessi sulla curva di domanda espressa a livello della microstruttura, variando in funzione del prezzo delle attività e del rendimento atteso.

4.2.1 La struttura della simulazione

Le scelte possibili in termini di attività finanziarie per gli agenti sono costituite da due tipi di strumenti che si distinguono per un differente rapporto

[1] Scritta nell'ambiente Swarm Objective-C.

in termini di profilo di rischio/rendimento:

- la prima attività disponibile è costituita da un titolo *risk free* disponibile in quantità infinita in grado di pagare un tasso di interesse costante (r_f);
- la seconda da un titolo rischioso, remunerato con un dividendo stocastico determinato secondo il modello autoregressivo che segue:

$$d_t = d + \rho(d_{t-1} - d) + \mu$$

con

$$\rho = 0.99$$

$$\mu_t \sim N(0, \sigma_\mu^2)$$

Lasciando libertà allo sperimentatore di variare i principali parametri della simulazione, gli autori hanno ipotizzato che il numero delle attività rischiose sia uguale al numero degli agenti. Altra ipotesi è che tutta la parte di ricchezza non investita nel titolo rischioso sia destinata ad essere impiegata nel titolo *risk free*. Inoltre, in ogni periodo gli agenti non potranno negoziare più di dieci titoli con il limite di cinque in caso di vendite allo scoperto, senza possibilità di indebitarsi.

Gli agenti sono avversi al rischio: sull'orizzonte temporale, limitato ad un periodo, massimizzano una funzione di utilità con avversione assoluta al rischio costante (CARA), del tipo:

$$E_t^i(-e^{-\gamma W_{t+1}^i})$$

con il vincolo:

$$W_{t+1}^i = x_t^i(p_t + 1 + d_t + 1) + (1 + r_f)(W_t^i - p_t x_t^i)$$

dove: E_t^i è la previsione dell'agente i al tempo t; γ rappresenta il coefficiente di avversione assoluta al rischio; $W_t + 1^i$ costituisce la ricchezza dell'agente i al tempo $t + 1$; x_t^i rappresenta la domanda di attività rischiose espressa dall'agente i al tempo t che, ipotizzando una distribuzione normale di prezzi e dividendi, è data da:

$$x_t^i = \frac{E_t^i(p_{t+1} + d_{t+1}) - (1 + r_f)p_t}{\gamma \sigma_{p+d,i}^2}$$

con $\sigma_{p+d,i}^2$ stima dell'agente i sulla varianza dei rendimenti $(p + d)$.

La funzione di domanda definita in questo modo è rappresentativa solo in casi di una distribuzione dei rendimenti normale, ipotesi alla base della teoria delle aspettative razionali. Distribuzioni dei rendimenti asimmetriche

invalidano il legame tra la funzione di avversione al rischio con γ costante e la funzione di domanda definite in precedenza.

Se si ipotizza una relazione lineare tra il prezzo dell'attività rischiosa e il dividendo come la seguente:

$$p_t = f d_t + e$$

data la costanza di γ per tutti gli agenti, è possibile trovare l'equilibrio di mercato, inserendo l'ultima relazione nella funzione di domanda e imponendo ad ogni agente di detenere una unità rischiosa in ogni periodo:

$$f = \frac{\rho}{1 + r_f - \rho}$$

$$e = \frac{d(f+1)(1-\rho) - \gamma\sigma^2_{p+d}}{r_f}$$

Sulla base della linearità che caratterizza la relazione tra p e d è possibile determinare le previsioni ottimali ad aspettative razionali in equilibrio; ció è possibile con l'equazione seguente che rappresenta una delle relazioni oggetto di stima da parte degli agenti:

$$E(p_{t+1} + d_{t+1}) = \rho(p_t + d_t) + (1-\rho)((1+f)d + e)$$

L'impostazione neoclassica del modello è evidente: gli agenti, caratterizzati da funzione di domanda e di avversità al rischio così definite, esprimono la loro domanda di attività rischiose sulla base delle proprie previsioni. A determinare l'equilibrio di mercato interviene un banditore (*specialist*); la sua azione assicura la formazione di un prezzo in grado di uguagliare domanda aggregata e offerta .

La novità del modello ASM è determinata dall'utilizzo delle tecniche della simulazione che superano il semplice aspetto descrittivo della teoria: grazie all'adozione del paradigma ad agenti è possibile ipotizzare una molteplicità di operatori, presenti nello stesso mercato, ognuno caratterizzato da una propria funzione di domanda espressione delle relative previsioni formulate.

Obiettivo dell'agente è quello di esprimere la propria domanda di attività rischiose al mercato, valutando previsioni sull'andamento futuro dei prezzi e dei dividendi. In questa attività di previsione, la struttura cognitiva dell'agente è costituita da un *classifier system*, impostato sulla base di una serie di relazioni di "condizione-previsione".

4.2.2 Le regole degli agenti

Ad ogni agente è assegnato un determinato insieme di regole (pari a 60) di classificazione (*classifier rules*) che descrivono, sulla base di un sistema binario, le possibili condizioni di mercato; sulla base degli strumenti cognitivi

a disposizione, l'agente valuta la corrispondenza della condizione attuale di mercato con una possibile regola di previsione al fine di stimare prezzo e dividendo del periodo successivo. Le possibilità di scelta dell'agente sono estese dall'ipotesi che più regole possano soddisfare una specifica condizione del mercato e quindi possano potenzialmente essere attivate per determinare la stima.

Utilizzando le tecniche degli algoritmi genetici, l'accuratezza della previsione, e di riflesso la regola attivata, verranno monitorate al fine di valutare l'aderenza con la reale dinamica di prezzo e dividendo. Grazie a questa valutazione e ad un processo di apprendimento ripetuto, il sistema cognitivo dell'agente sarà in grado di evolversi selezionando le regole migliori[2] da attivare e da utilizzare per la creazione di nuovi schemi di comportamento.

Il set di regole durante la simulazione viene modificato in corrispondenza di un ciclo diverso (ogni k periodi) per ogni agente che compone la popolazione: ciò comporta che in ogni periodo della simulazione vi sia un solo agente che rivede i propri schemi di comportamento grazie all'apprendimento. Durante questa fase di correzione delle regole, gli agenti modificheranno il proprio set, eliminando il 10% della regole capaci di fornire le prestazioni peggiori e sostituendole con le nuove ottenute dall'evoluzione dell'algoritmo genetico. Le operazioni "genetiche" saranno frutto dalla mutazione[3] e del *cross-over*[4] uniforme del restante 90% delle regole sopravvissute.

Gli agenti del modello hanno nel proprio set di regole, espresso in forma binaria da un vettore di sedici elementi, sia regole di analisi tecnica[5] sia di analisi fondamentale[6]. Tuttavia, i risultati prodotti dall'attivazione o meno di queste regole non fornisce alcun elemento addizionale al modello delle aspettative razionali in quanto la stima di prezzo e dividendo è basata esclusivamente su relativi valori correnti.

Ad inizio della simulazione, le regole sono inizializzate casualmente; questa scelta è funzionale alla volontà di costituire un insieme di regole il più generale possibile, totalmente indipendente dalle condizioni iniziali. Agli agenti è attribuita una ricchezza pari ad una attività rischiosa e 10000 unità di moneta.

[2]Valutate sulla base di una misura di "forza" relativa, capace di penalizzare le regole complesse e di assicurare che le informazioni utilizzate siano realmente utili in termini previsionali.

[3]Operatore con cui,nell'impostazione originale, viene modificato il carattere interno della regola genitrice con probabilità pari al 1%.

[4]Operatore che determina combinazione del patrimonio genetico di due regole genitrici, scelte tra quelle superstiti della popolazione iniziale mediante il meccanismo della *tournament selection*, con un probabilità del 30% (parametro questo opportunamente modificabile dallo sperimentatore). Per ogni nuova regola creata, la *tournament selection* ne determina la selezione di due, con utilizzo solo della più "forte" tra queste. La nuova regola eredita l'accuratezza della previsione media dei due genitori.

[5]Derivanti dal confronto del prezzo con una media mobile di varia lunghezza.

[6]Derivanti dalla valutazione del rapporto $prezzo * \frac{interesse}{dividendo}$.

Individuata la regola da attivare ed applicare, l'agente sostituisce i parametri, frutto della previsione, all'interno della propria funzione di domanda di attività rischiosa. Aggregate tutte le domande espresse dagli agenti, il prezzo dell'attività deriva dalla confronto con l'offerta fissata. Una volta avvenuta la transazione, gli operatori saranno in grado di verificare l'accuratezza della regola previsionale utilizzata attraverso il confronto con una opportuna media ponderata, correlata alla lunghezza dell'orizzonte temporale considerato nella propria attività di stima di prezzo e dividendo.

Valori elevati nella lunghezza dell'orizzonte temporale di valutazione indicano l'ipotesi implicita, nello schema di stima dell'agente, di una sorta di sostanziale "stabilità" nelle condizioni di mercato, mentre valori ridotti di questo parametro implicano la percezione di un mercato in rapida evoluzione. Nel caso limite di agenti che basano le loro stime solo sull'ultimo errore di previsione, le regole si fondano essenzialmente sul "noise" del mercato.

4.2.3 Aspetti tecnici della applicazione in Swarm

Il modello ASM è stato realizzato con l'utilizzo di Swarm, sfruttando proprietà e vantaggi offerti da questo strumento di sviluppo di applicazioni ad agenti.

L'applicazione, basata sulla definizione di dodici classi[7], può essere gestita dall'utilizzatore agendo sui parametri del modello attraverso i pannelli di controllo del ASMmodelSwarm e del ASMobserverSwarm. Dall'ASMmodelSwarm è possibile definire e modificare le variabili relative alle caratteristiche strutturali del modello attraverso le seguenti *probe*:

- *numBFagents*: il numero degli agenti dell'esperimento;

[7] Così definite:

- *Agent*: contiene le variabili e funzioni comuni a tutti gli agenti;
- *Dividend*: controlla il processo autoregressivo che genera il dividendo;
- *World*: gestisce e registra informazioni su regole attivate e valori dei dividendi e dei prezzi;
- *Specialist*: gestisce la fase di contrattazione dei titoli;
- *BFAgent*: contiene la descrizione dell'agente base del modello (*Bit-string-Forecasting agent*);
- *BitVector*: crea la struttura di bit necessaria alla simulazione;
- *BFCast*: è la classe dotata dei metodi che consentono di manipolare le informazioni;
- *BFParams*: gestisce i parametri utilizzati dagli agenti durante la simulazione;
- *ASMModelSwarm*: definisce il model dell'esperimento;
- *ASMObserverSwarm*: definisce l'observer del modello;
- *Output*: controlla la gestione dei dati e dei parametri su file;
- *ASMBatchSwarm*: lancia la simulazione con finalità di controllo.

- *initholding*: la dotazione iniziale di attività rischiose;
- *initcash*: la dotazione iniziale di titoli *risk free*;
- *minholding*: il limite di vendite allo scoperto;
- *mincash*: il limite di indebitamento;
- *intrate*: il tasso di interesse sull'impiego esente da rischio r_f;
- *mindividend, maxdividend, minprice, maxprice*: i minimi e massimi per dividendi e prezzi;
- *baseline*: il valore centrale attorno al quale viene calcolato il dividendo;
- *amplitudine*: l'ampiezza delle deviazioni da *baseline*;
- *period*: il periodo di autocorrelazione del processo;
- *exponentialMAs=1, exponentialMAs=0*: il tipo di media utilizzare nel calcolo dell'accuratezza delle regole; una media mobile pesata esponenzialmente (1) o una semplice media mobile di n periodi (0);
- *tauv*: il parametro τ che definisce l'orizzonte temporale preso in considerazione dall'agente;
- *RE = 0, SP = 1, ERA = 2*: il tipo di banditore[8] da utilizzare;
- *taup, eta, etamin, etamax, maxiterations, minexcess, rea, reb*: i parametri da utilizzare nel calcolo del prezzo di equilibrio;
- *randomseed*: l'innesco dei numeri casuali (randomseed = 0 indica l'innesco casuale);
- *maxbid*: l'offerta massima di attività rischiose;
- *initvar*: la varianza iniziale;
- *maxdev*: la massima deviazione di una previsione nella stima della varianza;
- *lambda*: il coefficiente di avversione al rischio.

[8] Vi sono 3 differenti tipi di banditore (*specialist*):

- lo *specialist RE*, il cui scopo è di controllare se la simulazione raggiunga l'equilibrio nel caso di aspettative razionali;
- lo *specialist SP* che calcola il prezzo di equilibrio tra domanda e offerta;
- lo *specialist ETA* che calcola il prezzo di equilibrio usando un processo di calcolo basato su di un parametro di stima η.

Con l'interfaccia grafica dell'ASMobserverSwarm è possibile, invece, impostare i parametri di visualizzazione e di gestione dei dati della simulazione attraverso i seguenti strumenti:

- *displayFrequency*: la frequenza di visualizzazione degli oggetti grafici;
- *writeSimulationData*: consente di salvare su file i dati di una particolare applicazione;
- *writeSimulationParams*: consente di salvare i parametri della simulazione.

Particolarmente importante risulta la gestione della scansione degli eventi che costituiscono la simulazione.

Al tempo t_0, il modello inizializza i propri parametri ed esegue una serie di cicli di l'apprendimento per le strutture cognitive degli agenti. Successivamente, al tempo t_1, la simulazione compie una sequenza di step, destinati a ripetersi anche in t_2 e negli istanti successivi:

1. si determina il dividendo del periodo precedente;
2. avviene l'aggiornamento della struttura informativa e dello stato del *World*[9];
3. lo *specialist* riceve le domande degli agenti, formulate sulle previsioni elaborate al periodo precedente, e determina il prezzo di equilibrio di mercato e, implicitamente, i dividendi reali;
4. gli agenti, avuta conoscenza di prezzo e dividendi effettivamente determinati dal mercato, provvedono ad aggiornare le proprie regole di previsione e la valutazione della loro ricchezza sulla base del rendimento dell'attività rischiosa e di quella priva di rischio;
5. grazie alle nuove informazioni, gli agenti utilizzano la dotazione di regole di previsione per stimare prezzo e dividendo del periodo successivo.

Una attenta analisi del modello è stata compiuta da LeBaron (2000) sulla base dei risultati di 25 diverse esecuzioni, caratterizzate da diversi parametri strutturali della simulazione. In particolare è stata posta specifica attenzione all'innesco dei numeri casuali e alla frequenza di apprendimento, identificando per quest'ultimo fattore due distinti valori ($k = 250$ e $k = 1000$).

Ogni esecuzione della simulazione ha avuto 250 mila cicli di apprendimento mentre le serie storiche utilizzate per l'analisi sono state osservate e registrate per i successivi 10 mila.

[9]Istanza della classe omonima in grado di conservare e gestire stringhe di valori corrispondenti allo stato del mercato e in grado di registrare le informazioni di prezzi e dividendi.

4.2.4 Conclusioni

Dall'analisi delle serie storiche risultanti, si è osservato come vi siano periodi nei quali il prezzo teorico di equilibrio, calcolato con l'impostazione delle aspettative razionali, non si discosta in maniera significativa dal prezzo effettivamente scambiato sul mercato. Tuttavia, accanto a questi periodi di stabilità della relazione ve ne sono altri dove la differenza è decisamente più sensibile: in qeeste fasi sembra che la valutazione delle attività finanziarie avvenga non secondo il paradigma delle aspettative razionali, ma secondo altri criteri.

L'analisi econometria delle serie storie generati dall'applicazione ripetuta del modello ha evidenziato il ruolo determinante in queste dinamiche svolto dalla frequenza di apprendimento. Con sequenze di apprendimento lento ($k = 1000$), il comportamento degli agenti appare coerente con gli schemi della teoria delle aspettative razionali e le serie storiche dei prezzi scambiati non si discostano da quelle teoriche. Nel caso, invece, di apprendimento veloce, con $k = 250$, le serie presentano notevoli scostamenti da quelle attese ed assumono andamenti simili a quelli osservabili nella realtà dei mercati finanziari. Inoltre, l'utilizzo dell'apprendimento veloce implica un ammontare di attività scambiate approssimativamente pari al doppio di quello negoziato con l'apprendimento lento ed un maggior uso dell'informazione derivante della componente legata all'analisi tecnica.

Analizzando l'andamento di volumi e volatilità si riscontra un legame positivo tra le due grandezze, frutto della notevole dispersione delle stime degli agenti in condizioni di mercato caratterizzato da una forte variabilità nei rendimenti. L'effetto erratico aggregato che ne risulta è in grado di più che compensare la diminuzione della volontà di riallocazione del proprio portafoglio di attività in situazioni di elevata incertezza, osservabile in ottica micro.

Un altro effetto, posto in luce da LeBaron, deriva dalla scarsa risposta nell'aggiornamento nelle regole di previsione nel caso di apprendimento lento che può essere indicata come possibile causa nella stabilità dei parametri di stima: questi parametri, simili in entrambe le configurazioni di apprendimento, tendono a quelli dell'equilibrio ad aspettative razionali.

L'analisi del modello pone in evidenza gli effetti svolti dal parametri legati alla frequenza nell'apprendimento. La conclusione a cui giunge LeBaron ne ricerca gli effetti e le implicazioni sui meccanismi che guidano la formazione delle regole di comportamenti così differenti. L'analisi del comportamento dell'algoritmo genetico evidenzia come un apprendimento lento implichi una scarsa evoluzione delle regole di formazione delle aspettative: le regole che ne derivano rispondono a schemi semplici e i prezzi, frutto di scambi limitati, che ne derivano sono sostanzialmente stabili. Il mercato sembra essere in grado di raggiungere l'equilibrio, l'"ottimo", ipotizzato dall'ipotesi di aspettative razionali.

Al contrario, in caso di apprendimento veloce si ha una maggior evoluzione delle regole di previsione da parte dell'algoritmo genetico, capace di rispondere più rapidamente alle variazioni del mercato che in questo caso non raggiungerà situazioni stabili. La maggior capacità di risposta degli agenti ha l'effetto di aumentare sensibilmente la varianza dei rendimenti, accentuando ulteriormente le implicazioni in termini di dispersione delle stime sull'andamento dei volumi scambiati.

Tale accentuata reattività, accompagnata da un maggior uso delle informazioni provenienti dall'uso delle informazioni derivanti dall'analisi tecnica, non si riflette in un corrispondente incremento della ricchezza accumulata dagli agenti che risulta inferiore rispetto al caso di apprendimento lento.

Queste conclusioni rendono evidente l'importanza dell'orizzonte temporale degli agenti che operando in mercati volatili e scarsamente prevedibili tenderanno a rivedere frequentemente i propri schemi di comportamento, abbandonando le assunzioni di razionalità alla base della impostazione neoclassica. I loro schemi, influenzati da indicazioni di tipo "tecnico", saranno maggiormente sensibili a variazioni di breve nei prezzi, con l'effetto di aumentare significativamente la variabilità dei rendimenti. Per contro, l'adozione di orizzonti temporali più ampi, ha l'effetto di spingere gli agenti verso schemi di valutazioni dipendenti dalle valutazioni di tipo "fondamentale", riducendo la volatilità dei mercati che tenderanno all'equilibrio ipotizzato dal paradigma delle aspettative razionali.

4.3 Il modello *Genoa artificial market*

Il *Genoa artificial market* costituisce un altro esempio di simulazione ad agenti applicata ad un mercato finanziario. Il modello, descritto in Raberto ed altri (2001), presenta caratteristiche peculiari derivanti dall'introduzione di ipotesi realistiche sulle risorse limitate a disposizione di ogni agente. Se per ASM l'obiettivo era principalmente quello di dimostrare le possibilità offerte dalla simulazione ad agenti nell'ambito dello studio dei mercati finanziari, gli autori del *Genoa market* mirano a costruire un robusto modello multiagente[10] con il quale sia possibile sviluppare esperimenti computazionali usando vari tipi di operatori artificiali.

4.3.1 La struttura del modello

Il modello rappresenta un mercato finanziario artificiale costruito con la tecnica ad agenti in cui una molteplicità di operatori scambia una sola attività sulla base di un meccanismo realistico di negoziazione per la formazione del prezzo.

[10] Basato sulle tecniche di programmazione ad oggetti.

Il processo di formazione dei prezzi nel mercato è costituito sulla base del meccanismo di corrispondenza tra domanda ed offerta, punto cardine del modello. In maniera del tutto realistica, gli autori hanno assunto che le risorse degli agenti siano limitate e che, nell'orizzonte temporale considerato per la negoziazione, l'ammontare globale di danaro presente nell'economia sia costante. Non sono previsti meccanismi di creazione del danaro: gli agenti sono vincolati a scambiare una sola attività sul mercato in cambio di contanti.

In ogni periodo gli agenti effettuano decisioni casuali di acquisto o vendita che sono vincolate dalle risorse disponibili e che dipendono dalla volatilità del periodo precedente. L'adozione di operatori con risorse limitate crea notevoli vincoli nel processo di decisionale dell'agente e non poche difficoltà agli autori in fase di sviluppo del modello. L'esecuzione ripetuta della simulazione in cui gli agenti così definiti erano suddivisi in gruppi caratterizzati da differenti strategie operative di scambio, basate su schemi dell'analisi tecnica, dell'analisi fondamentale e da regole casuali, ha posto in luce come, ad ogni esperimento, un gruppo di operatori finisse per prevalere, facendo perdere ricchezza e importanza agli altri. Una possibile soluzione a questo problema poteva essere rappresentata dalla possibilità di consentire passaggi casuali tra i membri delle popolazioni, ma questo assunto, con l'implicazione di spingere alcuni agenti ad adottare strategie perdenti, è sembrato poco realistico. Ciò ha portato gli autori a scegliere di mantenere semplici gli operatori e di dotarli di schemi decisionali di negoziazione basati sulla mera casualità: gli ordini sono generati in maniera totalmente casuale.

L'applicazione del modello ad una popolazione di agenti operanti in modo causale in un mercato caratterizzato da risorse limitate ha generato processi di prezzo caratterizzati da una distribuzione normale e comportamenti *mean reverting*; sono risultati assenti invece distribuzioni con code spesse e cluster di volatilità. Per rappresentare più fedelmente i processi di prezzo del mondo reale, si è resa necessaria l'introduzione di meccanismi capaci di riprodurre meglio i comportamenti reali degli operatori.

Sono stati introdotti e modellati fenomeni in grado di riprodurre struttura di gruppo[11] e un collegamento[12] tra la volatilità del mercato e l'incertezza degli agenti. L'applicazione di questi meccanismi in un processo di formazione di prezzi di mercato mostra comportamenti "reali" in termini di ispessimento delle code, di assenza di autocorrelazione dei rendimenti e di periodica autocorrelazine della volatilità.

[11] Creando aggregazioni di agenti come cluster.

[12] Ottenuto grazie al meccanismo di inserimento degli ordini da parte degli agenti, sulla base dell'assunto che in mercati volatili aumenta l'incertezza degli operatori sul prezzo delle attività finanziarie. Per rappresentare questo, gli ordini sono sì causali, ma presentano limiti di prezzo dipendenti dalla volatilità storica dei prezzi.

4.3.2 La microstruttura del Genoa market

Nel modello *Genoa market* gli agenti, in ogni *step*[13] della simulazione, inviano al sistema ordini di acquisto con probabilità P_i (valore impostato a 0.5), dove con i si indica l'agente *i-esimo*, oppure ordini in vendita con probabilità $1 - P_i$.

In caso di un ordine di vendita, la quantità offerta, a_i^s, per il tempo successivo $h+1$ è una frazione causale della quantità dell'attività finanziaria, indicata con $A_i(h)$, detenuta al tempo h; questo in base alla funzione, che restituisce un intero:

$$a_i^s = [r_i A_i(h)]$$

dove r_i è un numero casuale tra $[0, 1]$ estratto da una distribuzione normale. Ad ogni ordine di vendita è associato un prezzo limite[14], espresso da s_i, derivante da:

$$s_i = \frac{p(h)}{N_i(\mu, \sigma_i)}$$

dove $\mu = 1.01$ e σ_i sono media e standard deviation di una distribuzione normale $N_i(\mu, \sigma_i)$. La determinazione di σ_i è proporzionale alla volatilità storica del prezzo $p(h)$, indicata con $\sigma(T_i)$ e pari alla *standard deviation* dei rendimenti dei prezzi nell'orizzonte temporale T_i, secondo l'equazione:

$$\sigma_i = k\sigma(T_i)$$

con k costante, stimata dagli autori del modello ad un valore pari a 3.5. L'aver collegato la determinazione degli ordini con prezzo limite alla volatilità, ha introdotto nel modello un aspetto realistico della psicologia del *trading*: quando la volatilità è alta, cresce l'incertezza sui valore reale del titolo spingendo gli operatori a inserire nel sistema ordini con una distribuzione dei prezzi limite più ampia.

Gli ordini in acquisto ricalcano in maniera speculare molti degli aspetti che caratterizzano quelli in vendita. L'ammontare di danaro impiegato nell'ordine di acquisto, indicato con c_i, per il tempo $h + 1$ è una funzione causale della disponibilità liquida al tempo h.

$$c_i = r_i C_i(h)$$

dove con $C_i(h)$ si indica l'ammontare di danaro al tempo h. Anche per gli ordini in acquisto esiste un prezzo limite, b_i, per quotazioni al di sopra del quale l'ordine non può essere eseguito pari a:

$$b_i = \frac{p(h)}{N_i(\mu, \sigma_i)}$$

dove i parametri sono quelli indicati in precedenza.

[13] Si assume che il tempo, indicato con h, trascorra in maniera discreta.

[14] Al di sotto del quale l'ordine di vendita non può essere eseguito.

La quantità di attività finanziaria in acquisto, a_i^b, sarà pari all'intero restituito dalla equazione :

$$a_i^b = [\frac{c_i}{b_i}]$$

L'aver assunto $\mu = 1.01$ implica che la media dei valori di b_i e la media dei valori di s_i siano rispettivamente maggiore e minore di $p(h)$. Lo *spread* introdotto tra la media dei due prezzi si caratterizza dall'ipotesi realistica che ogni operatore, quando inserisce un ordine, voglia accrescere la probabilità che venga eseguito.

Essendo r_i e $N_i(\mu, \sigma_i)$ generati indipendentemente in ogni step per ogni agente, ciò implica che ogni operatore presenti un comportamento casuale eterogeneo, sottoposto a due vincoli: la volatilità storica, inclusa in σ_i, e le risorse limitate disponibili per l'individuo.

Il processo di formazione del prezzo è determinato dalla intersezione delle curve di domanda ed offerta calcolate nel modo seguente. Si supponga che al tempo $h + 1$, gli agenti abbiamo espresso U ordini di acquisto e V ordini in vendita. Per ogni ordine di acquisto, la coppia (a_u^b, b_u) con $u = 1, 2, \ldots, U$ indica, rispettivamente, la quantità di attività in acquisto e il relativo prezzo limite. Analogamente per ogni ordine di vendita allo stesso istante, la coppia (a_v^s, s_v) con $v = 1, 2, \ldots, V$, indica la quantità di titoli in vendita e il relativo prezzo limite.

Si introducono allora le funzioni:

$$f_{h+1}(p) = \sum_{u|b_u \geq p} a_u^b$$

$$g_{h+1}(p) = \sum_{u|s_u \leq p} a_v^s$$

dove $f_{h+1}(p)$ rappresenta la quantità totale delle azioni in acquisto al prezzo p, la funzione di domanda decrescente al crescere del prezzo; mentre $g_{h+1}(p)$ rappresenta l'ammontare totale di azioni in vendita al prezzo p, la funzione di offerta con proprietà simmetriche a quelle della domanda.

Il prezzo di equilibrio calcolato dal sistema sarà il prezzo $p*$, livello di intersezione delle due curve. Il nuovo prezzo espresso al tempo $h + i$ dal mercato sarà: $p(h + 1) = p*$. In corrispondenza di questo prezzo, la quantità domandata di titoli, indicata con $f(p*)$, sarà pari alla somma di tutte le proposte espresse con un prezzo limitato maggiore o uguale a quello di equilibrio; la somma delle proposte in vendita con prezzo limite inferiore o uguale a quello di equilibrio costituirà la quantità offerta che sarà pari $g(p*)$. Essendo le due funzioni discrete, gli autori del modello hanno dovuto introdurre opportune semplificazioni per rendere uguale in corrispondenza del prezzo $p*$ le quantità espresse dalle due curve. Con questa semplificazione, ordini in vendita e in acquisto hanno la stessa quantità e possono essere eseguiti. In seguito alle transazioni, i portafogli di attività degli operatori sono

aggiornati e gli ordini che non hanno trovato contropartita in corrispondenza del prezzo di equilibrio sono cancellati.

Infine gli autori hanno rappresentato il processo di diffusione delle stime tra gli agenti grazie alla tecnica di costruzione dei *cluster*. In ogni *step* di tempo, coppie di operatori vengono scelte in maniera casuale con probabilità P_a: tra gli elementi della coppia scelta si viene a formare un *cluster*. Iterando il processo con il trascorrere del tempo, si formeranno progressivamente aggregazioni di operatori destinate a crescere ed eventualmente ad unirsi fra loro.

In ogni passo della simulazione, viene estratto un numero casuale in grado di scegliere se attivare (con probabilità P_c) un *cluster*, scelto a caso tra tutti quelli del sistema, oppure se lasciare questi ultimi inattivi (con probabilità $1 - P_c$). Nel caso si decida di attivare l'aggregazione, scelta tra quelle della popolazione, tutti gli operatori che ne fanno parte vedono modificato il loro parametro P_i che passa dal valore di 0.5 a 1.0 o 0.0 con probabilità del 50%. In questo modo, tutti gli agenti del *cluster* avranno lo stesso comportamento, inserendo ordini di acquisto se P_i sarà pari a 1 o in vendita se sarà 0. Una volta inseriti gli ordini, il *cluster* verrà disaggregato e i valori di P_i degli operatori che ne facevano parte torneranno ad essere pari a 0.5.

Come accennato in precedenza, il *Genoa market* introduce una ulteriore correzione per tener conto della volatilità dei prezzi: gli ordini con limite sono correlati con questa grandezza. Ciò permette due osservazioni:

a) sebbene l'ammontare di moneta e il numero delle azioni sono costanti nel tempo, la ricchezza complessiva è variabile essendo una funzione lineare del prezzo delle azioni. Il meccanismo di formazione dei prezzi crea o distrugge ricchezza;

b) le condizioni di risorse limitate imposte dagli autori alimentano un meccanismo di adeguamento del prezzo alla media. Si supponga che il prezzo del titolo salga al di sopra del livello di equilibrio in corrispondenza del quale le disponibilità monetarie eguagliano il valore complessivo dei titoli. Sulla base delle ipotesi introdotte nel modello, lo squilibrio tra ordini in acquisto e ordini in vendita tenderà a far tornare il prezzo verso il valore di equilibrio.

In questo modo, si crea un ulteriore effetto di retroazione che tende ad uniformare le distribuzioni di titoli e moneta tra gli agenti: gli operatori con maggiori quantità di azioni tenderanno ad inserire più ordini in vendita che in acquisto e vice versa per gli agenti più "liquidi".

4.3.3 Risultati e conclusioni della simulazione

Gli autori hanno effettuato una serie di simulazioni al termine della quali hanno ottenuto risultati in termini di distribuzioni dei rendimenti logaritmici

simili a quelli ottenuti dall'osservazione empirica di varie serie di prezzi di attività finanziare reali con frequenza giornaliera.

Analizzando le autocorrelazioni dei rendimenti logaritmici standardizzati in valore assoluto e dei rendimenti grezzi con diversi *lag* temporali, si è osservata una rapida diminuzione per i dati grezzi ed una relativa stabilità delle autocorrelazioni per i dati in valore assoluto. Assumendo i rendimenti in termini assoluti come una misura della volatilità, le serie simulate hanno mostrato un comportamenti simili a quelli presenti sui mercati del mondo reale, mettendo in evidenza la formazione di *cluster* di volatilità. Si è anche osservato che la volatilità è sensibile alla scala del modello: al crescere del numero degli agenti, diminuisce la varianza dei rendimenti.

In definitiva, il modello si è dimostrato capace di riprodurre la forma leptocurtica della densità di probabilità mostrata dai log-rendimenti dei prezzi e di rappresentare i cluster di volatilità tipici dei mercati finanziari. Questo risultato conferma le ipotesi e le conclusioni di un precedente lavoro di uno degli autori (Chen e altri, 2001) con cui è stato possibile mostrare, sempre all'interno di un mercato simulato, l'emergere endogeno di dinamiche nonlineari dai processi di negoziazione di attività finanziarie e dalle interazioni tra gli agenti.

Nonostante il modello *Genoa Market* non sia ancora riuscito a rappresentare altri aspetti della dinamica dei prezzi, come ad esempio la diminuzione della volatilità secondo una funzione potenza[15], tuttavia, costituisce un laboratorio sperimentale, tuttora in fase di sviluppo, su cui svolgere esperimenti computazionali di simulazione dei mercati finanziari.

[15]Le serie prodotte dal modello evidenziano una diminuzione della volatilità in legge esponenziale, in disaccordo con l'evidenza empirica del mondo reale; a questo riguardo occorre sottolineare che dagli studiosi non è ancora stata fornita una teoria generalmente accettata del fenomeno e che l'applicazione dei modelli econometrici ARCH e GARCH, applicati alle correlazioni della volatilità in differenti istanti di tempo, ha mostrato diminuzioni della volatilità secondo leggi esponenziali.

Capitolo 5

Da SUM a SUMWeb

Il modello SUM, *Surprising (Un)realistic Market*, descritto in questo capitolo rappresenta la simulazione di un mercato finanziario basata su agenti, partendo dalla ricreazione delle sue microfondazioni. Il modello, sviluppato in Terna (2000), si è mostrato capace di spiegare il processo di formazione dei prezzi tipico di questo tipo di mercato, eliminando alcune delle ipotesi semplificatrici introdotte in ambito classico quali, ad esempio, quella dell'introduzione di un banditore in grado di far da compensazione al mercato. La simulazione, invece, ricrea il funzionamento di un mercato finanziario reale formato da agenti eterogenei che esprimo le loro specifiche proposte di negoziazione per un'unica attività: sarà l'abbinamento di ordini di segno opposto, in perfetta analogia con quanto avviene sul mercato reale, a determinare la successione dei prezzi, in continua evoluzione, transazione dopo transazione.

L'aspetto chiave del modello è rappresentato da un struttura computazionale che riproduce fedelmente regole e meccanismi del *book* di un reale mercato di borsa telematico. Questo strumento consente l'immediata esecuzione degli ordini espressi dagli agenti che abbiano una controparte di segno opposto oppure registra segno e quantità delle proposte inviate al mercato, separandole in proposte di acquisto (in linguaggio tecnico "danaro") e di vendita ("lettera"), per abbinarle ad ordini futuri.

Il tempo, discreto, svolge un ruolo fondamentale nella simulazione e determina lo svuotamento del *book* all'inizio di ogni giornata in analogia con quanto avviene nel mercato reale.

Gli agenti del modello sono eterogenei e a differenziarli è la loro struttura cognitiva che si traduce in schemi di comportamento diversi tra le varie classi di operatori ricreate. Nel sistema sono presenti agenti *no mind*, dotati di limitate capacità computazionali, e agenti *mind*, capaci di adeguare i propri schemi di azione alle diverse condizioni del mercato.

Sin dalla sua impostazione più semplice, basata su agenti operanti interamente in maniera casuale, l'esecuzione della simulazione ha mostrato la

formazione di sequenze di prezzo crescenti o decrescenti caratterizzate da una rilevante volatilità. In perfetta analogia con l'osservazione empirica, nelle serie dei prezzi è stata osservata la formazione di bolle speculative e di *crash*, generate in maniera endogena, senza il ricorso a meccanismi di controllo e coordinamento esterni.

Queste conclusioni, indipendenti dalle condizioni iniziali della simulazione, sono state confermate anche dalle successive complicazioni inserite nel modello, rappresentate dall'introduzione di più tipologie di operatori soggette a diverse regole di valutazione e azione.

Il modello, quindi, rappresenta un valido strumento per osservare le regolarità emergenti dal comportamento di diversi e molteplici operatori, analizzandone gli aspetti in ottica dell'individuo e aggregata. Ciò che emerge è la struttura del mercato come sistema complesso. La dinamica dei prezzi è la conseguenza di innumerevoli decisioni prese sistematicamente da un elevato numero di investitori. Dall'interazione delle loro diverse strategie, risultato di diversi schemi di valutazione del mercato e di aspettative sull'evoluzione futura, emerge un sistema complesso, il cui andamento è difficilmente prevedibile.

All'utilizzatore del modello è concessa la facoltà di intervenire su parametri e condizioni che costituiscono gli aspetti strutturali della simulazione; ciò consente di valutare l'impatto di eventuali variazioni su specifiche problematiche empiriche, quali la volatilità del mercato e la prevedibilità delle serie di tempo, difficilmente analizzabili con altri strumenti.

Per ottenere questi risultati, la simulazione è stata realizzata utilizzando le tecniche della programmazione ad oggetti descritte nei capitoli precedenti: il codice, liberamente disponibile in rete *web*, è scritto in Swarm Objective-C, sfruttando la potenza computazionale e la flessibilità di programmazione offerta da questo strumento.

La seconda parte del capitolo descrive la nuova versione[1] del modello, SUMWeb (*SUM Web economic behaviour*), che rappresenta il tentativo di "aprire" il mondo artificiale verso l'esterno, consentendo anche ad agenti "umani" di prendere parte alla simulazione.

Le potenzialità di un simile strumento, autentico laboratorio sperimentale, potrebbero varcare i confini della scienza economica per abbracciare in un approccio interdisciplinare anche teorie psicologiche e sociologiche sui comportamenti del'uomo in ambito finanziario.

5.1 La microstruttura dei mercati finanziari

Nell'analisi dei mercati finanziari, lo studio dei singoli elementi che li costituiscono, delle microfondazioni, diventa sempre più importante. I mercati finanziari sono caratterizzati da precisi assetti istituzionali, e lo stu-

[1] Tuttora in sviluppo.

dio di queste strutture diventa imprescindibile nella comprensione del funzionamento a livello sistemico. Nei mercati operano attori che svolgono ruoli istituzionali ben definiti, risultato di precise logiche economiche alla base del comportamento: proprio l'analisi di queste componenti diventa determinante per riuscire a comprendere fisiologia e morfologia dei mercati organizzati.

Abbandonato il modello di mercato Walrasiano basato su agenti che operano in perfetta concorrenza, scelta la via proposta dalla simulazione ad agenti, occorre, innanzitutto, descrivere l'assetto istituzionale all'interno del quale far operare gli operatori simulati.

I mercati finanziari presentano una serie di peculiarità che li caratterizzano e permettono di individuare la classificazione seguente (basata su Barucci, 2000):

a) *Mercato guidato dagli ordini-Mercato guidato dai prezzi:* la distinzione deriva dalle modalità seguite dagli ordini per essere eseguiti dal mercato. Nel primo caso, gli ordini degli operatori giungono direttamente sul mercato e solo in un secondo momento un meccanismo di abbinamento delle proposte o un *market maker*[2] interviene a definire il prezzo di scambio. Nel secondo caso, invece, sul mercato operano particolari operatori, detti *dealer*)[3], che svolgono la funzione di fornire in continuo e in contemporanea un prezzo di acquisto, o *bid*, e un prezzo di vendita, o *ask*, a cui si impegnano, in condizioni specifiche, a rappresentare la contropartita negli scambi su determinate attività finanziarie;

b) *Mercato a svolgimento nel continuo-Mercato a svolgimento nel discreto:* nel primo tipo di mercati le contrattazioni e di conseguenza gli scambi avvengono nel continuo e gli ordini arrivano sul mercato per essere soddisfatti da meccanismi automatici o da *market maker*; nel secondo, invece, le contrattazioni avvengono nel discreto, in corrispondenza di intervalli di tempo predeterminati. Fanno parte di quet'ultimo tipo di mercato, tutti i sistemi di contrattazione basati su un meccanismo ad asta in forma *batch*, in cui è il *market maker* che de-

[2]Il *market maker* ha una funzione di controllo del mercato e si impegna a fissare un prezzo in cui si dichiara pronto ad effettuare il *clearing*delle proposte su una determinata attività finanziaria.

[3]In questa attività, i *dealer* offrono un servizio agli operatori del mercato, impegnandosi ad intervenire per soddisfare gli scambi. Un simile tipo di servizio implica due costi/rischi per questa tipologia di operatori:

- l'essere vincolato ad adottare scelte di portafoglio non ottimali;
- il dover sottostare alla *adverse selection* nei confronti degli agenti presenti sul mercato.

Per far fronte a questi costi e per remunerare la propria attività, i *dealer* applicano uno *spread* tra *bid* e *ask*.

termina il prezzo del titolo e in corrispondenza di questo si dichiara pronto ad intervenire sul mercato;

c) *Mercato consolidato-Mercato frammentato:* a distinguere le due tipologie è la possibilità che uno stesso titolo venga scambiato su un solo o su più mercati;

d) *Mercato con broker-Mercato con dealer:* i due tipi di mercati sono caratterizzati rispettivamente dalla presenza di soggetti che operano come intermediari per conto terzi (*broker*) o in conto proprio (*dealer*);

e) *grado di competizione tra market maker:* i mercati si distinguono per la presenza contemporanea o meno di più *market maker* che operano sugli stessi titoli.

In base alla clssificazione proposta, è possibile identificare il modello proposto dalla versione attuale di SUM come un mercato guidato dagli ordini, caratterizzato da un fase discreta ed una continua, e privo di intermediari finanziari.

Nel modello svolge un ruolo fondamentale il meccanismo di abbinamento delle proposte di scambio da parte degli operatori.

5.2 La struttura di SUM: il *book*

Prima di definire gli altri aspetti strutturali della simulazione occorre illustrare le le principali caratteristiche del meccanismo di abbinamento delle diverse proposte espresse dagli agenti, identificato come il cardine del modello.

Il *book* di SUM riproduce, nelle linee fondamentali e con qualche opportuna semplificazione, il funzionamento di quello del mercato telematico azionario (MTA)[4] italiano descritto nel regolamento[5] emanato da Borsa Italiana S.p.A.

Le modalità di negoziazione, originariamente seguite per la costruzione del modello, prevedevano per entrambi i mercati, quello reale e quello simulato, due fasi distinte di scambio delle attività finanziarie e di formazione dei prezzi delle stesse. Recentemente, per dare maggior rappresentatività al prezzo di chiusura delle contrattazioni, l'organo di gestione del MTA ha introdotto una terza fase di contrattazione che si affianca alle due precedenti:

[4]L'MTA è il comparto di mercato gestito da Borsa Italiana S.p.A. in cui si negoziano azioni, obbligazioni convertibili, *warrant*, diritti di opzione e azioni di OICR quotati in borsa.

[5]Disponibile gratuitamente al sito *www.borsaitalia.it* ed attualmente in vigore dal 1 luglio 2002.

l' "asta di chiusura"[6]. Tali recenti modifiche regolamentari sono mantenute in parte distinte dallo schema originale seguito per il modello e sono oggetto di uno specifico paragrafo in appendice.

Secondo lo schema di base del MTA, utilizzato per la creazione di SUM, l'operatività sul mercato telematico poteva avvenire durante le fasi di:

- "asta di apertura"[7];
- "negoziazione continua".

L'asta di apertura è a sua volta articolata nella fase di determinazione del prezzo teorico di apertura (*"pre-apertura*), in quella di validazione del prezzo teorico (*"validazione*) e in quella di conclusione dei contratti (*"apertura*).

Lo schema di Sum riprende queste modalità, ma introduce una serie di opportune semplificazioni la prima delle quali consiste nel limitare l'operatività due sole fasi, una di pre-apertura ed in una continua, prescindendo da altre suddivisioni temporali.

In entrambi i mercati oggetto di analisi, gli operatori esprimono la propria volontà negoziale attraverso opportune *proposte di negoziazione* le quali contengono le informazioni relative allo strumento finanziario da negoziare[8], alla quantità[9], al tipo di operazione, al conto utilizzato nell'operazione[10] nonché alle condizioni di prezzo. Le proposte di negoziazione possono essere immesse, modificate e cancellate dagli operatori sia nella fase di pre-asta che nella negoziazione continua. Qui si ravvisa una nuova semplificazione introdotta nel modello: per semplicità agli agenti simulati non è data facoltà di modificare o cancellare le proposte immesse.

[6]Modalità di negoziazione che prevede l'immissione, la modifica e la cancellazione di proposte di negoziazione in un determinato intervallo temporale (pre-asta) al fine della conclusione dei contratti in un unico momento futuro (chiusura) e a un unico prezzo (prezzo d'asta di chiusura).

[7]Modalità di negoziazione che prevede l'immissione, la modifica e la cancellazione di proposte di negoziazione in un determinato intervallo temporale (pre-asta) al fine della conclusione dei contratti in un unico momento futuro (apertura) e a un unico prezzo (prezzo d'asta di apertura).

[8]Nell'attuale versione di SUM limitato ad una sola attività.

[9]La recente abolizione dell'obbligo di inserire proposte per quantità pari o multiple al quantitativo (lotto) minimo dal regolamento del MTA ha aumentato il realismo della simulazione visto che, su entrambi i mercati, gli operatori possono scambiare anche un solo titolo senza necessità di multipli minimi alle quantità. L'abolizione di questi minimi ha eliminato la distinzione tra le proposte immesse dagli operatori dell'MTA sulla base delle quantità, esistente precedentemente all'approvazione delle ultime modifiche regolamentari: gli ordini degli operatori potevano avere per oggetto quantitativi pari o multipli del lotto minimo di negoziazione (cd. Proposte di negoziazione "intere") ovvero quantitativi inferiori al lotto minimo (cd. Proposte di negoziazione "spezzature"). Queste due tipologie di ordini prevedevano modalità diverse di immissione nelle varie fasi di operatività del mercato.

A rigore in Sum, tutti gli operatori immettono proposte di negoziazione sempre in quantità unitaria fatta salva la possibilità di entrare nel *book* con più ordini.

[10]Caratteristica assente in SUM.

Le proposte sono ordinate in modo automatico per lo strumento a cui si riferiscono sulla base del prezzo con ordinamento decrescente in caso di acquisto e crescente in vendita. In caso di parità di prezzo, il criterio adottato è quello della priorità temporale determinata dall'orario di immissione. Nel mercato reale, in caso di modifica della proposta che implichi un aumento del quantitativo o una variazione di prezzo viene persa la priorità temporale acquisita.

Le proposte di negoziazione durante la fase di pre-asta[11] del MTA, possono venire immesse con o senza limite di prezzo (proposte al "prezzo di asta") e possono essere specificate con quattro distinte modalità di esecuzione:

1. "valida fino alla cancellazione";
2. "valida fino alla data specificata";
3. "esegui e cancella";
4. "valida fino all'orario specificato" solo se immesse con limite di prezzo.

L'immissione senza limite di prezzo (al "prezzo di asta") fa assumere dinamicamente a queste proposte il prezzo al quale avrebbero maggiori possibilità di essere eseguite.

Anche durante la negoziazione continua, le proposte intere possono essere immesse con o senza limite di prezzo ("proposte al prezzo di mercato"). Le modalità che caratterizzano l'immissione degli ordini nella fase di pre-asta si ritrovano (con una sola eccezione procedurale per gli ordini con "validità fino alla data specificata") anche durante la negoziazione continua in cui una proposta intera può essere:

1. "valida fino alla cancellazione";
2. "esegui e cancella";
3. "esegui quantità minima specificata";
4. "tutto o niente";
5. "valida fino alla data specificata";
6. "valida fino all'orario specificato";

e in presenza di un prezzo limite si distinguono ancora le modalità:

- "esegui comunque";
- "esponi al raggiungimento del prezzo specificato".

[11] Coincidente con l'apertura per SUM, sviluppato quando la fase di asta coinvolgeva solo il momento della pre-apertura e non si aveva ancora la fase di asta di chiusura.

Una ulteriore possibilite modalità di inserimento degli ordini concessa agli operatori prevede la facoltà di esclusione della visualizzazione dell'intera quantità proposta in presenza di un prezzo limite.

L'organo di gestione del mercato, per tutelare l'efficienza degli scambi, fissa limiti percentuali di variazione massima dei prezzi e pertanto non viene consentita l'immissione di ordini aventi un prezzo superiore o inferiore a tali limitazioni.

Il mercato simulato, SUM, introduce ulteriori opportune semplificazioni a questo schema che non compromettono la fedeltà della rappresentazione. Come detto in precedenza ogni proposta presenta sempre quantità unitaria e un prezzo limitato. E' esclusa, per ora, la possibilità di ordini "al meglio". L'unica tipologia di ordini prevista è quindi quella con limite di prezzo ed è esclusa ogni possibilità di gestione di altre modalità di esecuzione, previste invece dal MTA.

Una altra importante semplificazione è costituita dall'ipotesi di assenza di meccanismi di limitazione alle oscillazioni dei prezzi, previsti sul mercato MTA con lo strumento della sospensione alle negoziazioni e con il rifiuto delle proposte eccessivamente divergenti da parametri di controllo[12].

Per contro, il modello estende le possibilità degli operatori artificiali rispetto a quelle degli operatori del MTA, consentendo ai primi la vendita allo scoperto dei titoli e il mantenimento di posizioni "corte".

Il regolamento del mercato artificiale prevede che, nella prima fase (apertura), avvenga l'immissione di ordini da parte degli operatori con un determinato grado di probabilità, che, tuttavia, non concorrono a determinare alcun prezzo teorico di apertura: questo nella simulazione sarà pari all'ultimo prezzo della seduta precedente. Gli ordini in questa fase non sono conclusi al prezzo di pre-apertura, come nell'MTA, ma vengono trasferiti alla successiva negoziazione continua. Questa scelta riflette l'esigenza di non iniziare la giornata senza proposte di negoziazione nel *book* e comporta, in questa fase, l'assenza di abbinamento delle proposte fra loro.

Per contro, durante la fase di pre-asta (di apertura e chiusura) sul MTA, viene calcolato e aggiornato in tempo reale, a titolo puramente informativo, il prezzo teorico di asta sulla base delle proposte immesse nel sistema.

Il prezzo teorico di asta risulta pari al prezzo a cui è negoziabile il maggior quantitativo dello strumenti finanziario. Qualora vi siano più prezzi che godono del requisito richiesto in termini di quantità viene introdotta una serie di criteri in grado di gestire le potenziali eccezioni:

- il primo di questi prevede che in caso di più prezzi che soddisfino il criterio del maggior numero di strumenti finanziari scambiati, si identifichi il prezzo teorico di asta come quello che produce il minor quantitativo non negoziabile relativamente alle proposte in acquisto o in vendita, aventi prezzi uguali o migliori rispetto a quello considerato;

[12]Per maggiore chiarezza si veda l'appendice.

- qualora anche questo criterio non favorisca l'individuazione del prezzo teorico di asta viene scelto quello che risulta più prossimo all'ultimo prezzo di riferimento;
- nel caso in cui dall'applicazione del criterio precedente risultino due prezzi equidistanti dal prezzo di riferimento, il prezzo teorico di asta risulta pari al maggiore dei due;
- anche nel caso in cui in acquisto e in vendita siano solo presenti proposte esclusivamente al prezzo di teorico di asta, per la determinazione del prezzo teorico si ricorre all'ultimo prezzo di riferimento;

Per contro, in caso di assenza di proposte di negoziazione in uno o entrambi i lati del mercato, in presenza di proposte con limite di prezzo e miglior prezzo in acquisto inferiore al miglior prezzo di vendita o in caso quantitativi minimi inferiori al lotto minimo negoziabile, il prezzo teorico di apertura non può essere determinato.

Il prezzo teorico così determinato al termine della fase di validazione viene considerato valido e assunto come prezzo di asta (di apertura e di chiusura) per la conclusione dei contratti se il suo scostamento dal prezzo di controllo[13] non supera una percentuale fissata. Se ciò accade viene riattivata la fase di pre-asta per un intervallo di tempo stabilito. Nel caso in cui non sia stato determinato un prezzo teorico di apertura al termine della fase di validazione, viene attivata la fase di negoziazione continua per le proposte, che vengono trasferite automaticamente con la priorità temporale delle proposte originarie. Queste proposte hanno prezzo pari a quello della proposta originaria se immessa con limite di prezzo, della migliore proposta con limite di prezzo presente sul mercato se si tratta di proposte al prezzo di asta e sempre per queste ultime prezzo pari a quello di riferimento se al termine della fase di pre-asta non sono presenti proposte con limite di prezzo.

Nella successiva fase di asta si ha la conclusione dei contratti al prezzo di apertura che derivano dall'abbinamento automatico delle proposte in

[13] In base all'ultimo regolamento disponibile, il prezzo di controllo della giornata di ciascun strumento finanziario è dato dal:

a) prezzo di riferimento, in asta di apertura;

b) prezzo di asta di apertura, durante la negoziazione continua; qualora non sia stato determinato un prezzo di asta di apertura, il prezzo di controllo è pari a quello della lettera a);

c) prezzo di asta di apertura, in asta di chiusura; qualora non sia determinato un prezzo di asta di apertura, il prezzo di controllo è pari a quello di cui alla lettera a).

Il prezzo di riferimento è pari al prezzo di asta di chiusura. Qualora questo prezzo non sia stato determinato allora il prezzo di riferimento è posto pari alla media ponderata dell'ultimo dieci per cento delle quantità negoziate.

acquisto che hanno prezzi uguali o superiori al prezzo di asta con quelle in vendita che hanno prezzi uguali o inferiori allo stesso prezzo. Il procedimento di abbinamento avviene secondo le priorità di prezzo e di tempo delle singole proposte e fino all'esaurimento delle quantità disponibili.

Le proposte che restano ineseguite in tutto o in parte vengono trasferite dal sistema alla fase di negoziazione continua come proposte con limite di prezzo. Questo risulta pari al prezzo limite indicato nella proposta originaria oppure al prezzo di apertura nel caso di proposte al prezzo di apertura.

La riproduzione realizzata con SUM semplifica questa fase, limitandosi ad introdurre una fase di apertura per l'introduzione di ordini ad inizio seduta. Eventuali apporti in termini di realismo allo schema potranno arrivare da futuri sviluppi del modello.

Nella fase della negoziazione continua, la conclusione dei contratti avviene mediante l'abbinamento automatico delle proposte di segno contrario presenti sul mercato, ordinate secondo i criteri di priorità precedentemente individuati sulla base delle quantità disponibili.

Nel caso di immissione di una proposta di acquisto con prezzo limite si procede all'abbinamento con le proposte di vendita aventi prezzo inferiore o uguale a quello della proposta immessa. Qualora, invece, avvenga l'immissione di una proposta senza limite di prezzo in acquisto, l'abbinamento individua come contropartita all'operazione le proposte di vendita con prezzo uguale al miglior prezzo di vendita esistente al momento della sua immissione. Per poter immettere una proposta senza limite di prezzo risulta indispensabile la presenza di almeno una proposta di negoziazione di segno contrario con limite di prezzo.

Nel caso di proposte di vendita valgono le modalità indicate in maniera totalmente speculare. Il prezzo dei contratti conclusi in questa fase è pari a quello della proposta avente priorità temporale superiore mentre la quantità eseguita è funzione della disponibilità di una controparte.

La proposta al prezzo di mercato eseguita in maniera parziale dà vita alla creazione di una proposta che rimane esposta con il prezzo dell'ultimo contratto concluso e la priorità temporale originaria. In caso di proposta con limite di prezzo eseguita in maniera parziale rimane nel sistema con prezzo e priorità di quella originaria. Quanto esposto descrive le modalità di formazione di conclusione dei contratti in negoziazione continua sul MTA e ricalca pienamente quelle di SUM, tenendo conto dell'assenza di proposte diverse da quelle con prezzo limite.

Anche per questi aspetti si guarda ai futuri sviluppi del modello per veder crescere il grado di realismo della simulazione.

5.3 Lo schema del modello

L'ambiente della simulazione è costruito sulla base dello schema ERA, definito nel capitolo quarto.

Gli agenti adottano relazioni indirette nel modello: non vi è interazione diretta tra loro, ma esprimono le proprie decisioni al *book*, che abbinando le proposte di negoziazione espresse dai diversi agenti, assicura il corretto funzionamento del mercato finanziario artificiale.

Nell'ambito della simulazione così descritta, lo schema ERA trova perfetta applicazione.

Gli agenti possono essere classificati a seconda delle capacità cognitive di cui sono dotati, in agenti *mind* e *no mind*: ciò ha riflessi sulla loro collocazione all'interno dello schema citato e sulle strutture a cui accedono per modificare i loro comportamenti.

Gli agenti dotati di una struttura cognitiva *no mind*, non dovendo modificare le proprie regole di azione, sono in collegamento con due soli oggetti: il *book* e il *Rule Master*. Per gli agenti *mind*, invece, esiste il collegamento con il generatore di regole -il *Rule Maker*- che consente loro il costante adeguamento delle regole alla base dell'azione in risposta al mutare delle condizioni ambientali.

5.4 La gestione del tempo della simulazione

La seduta di contrattazione è tecnicamente articolata nelle seguenti fasi:

- *cleaning*: le proposte di negoziazione immesse dagli agenti il giorno precedente vengono cancellate; il *book* viene "svuotato" e predisposto per una nuova seduta;

- preapertura: in questa fase gli agenti decidono se immettere o meno proposte di negoziazione nel *book*; tali proposte non sono destinate ad essere eseguite, ma verranno trasferite alla successiva fasa di contrattazione continua; le preapertura serve da un punto di vista tecnico esclusivamente per non far iniziare le contrattazioni con un *book* privo di proposte;

- negoziazione continua: in questa fase, gli agenti immettono le proposte di acquisto o vendita, frutto delle proprie regole di comportamento; tali proposte, in linea con le regole descritte per il loro abbinamento, potranno essere eseguite immediatamente in caso vi sia una diretta controparte; in caso contrario le proposte di negoziazione verranno registrate con la relativa priorità temporale con la possibilità di essere concluse in un momento successivo della seduta;

- *accounting*: al termine delle contrattazioni viene valutato il prezzo medio della seduta che serve agli agenti per il calcolo della loro ricchezza composta dalla liquidità inutilizzata e dalle azioni possedute.

Il *timing* della simulazione, composto da un numero di *tick* pari al numero degli agenti della popolazione, in ogni giorno di borsa è il seguente:

- t_0
  - si aggiorna il giorno corrente della simulazione;
  - avviene lo svuotamento del *book*;
  - gli agente dotati di capacità previsionale formulano le proprie stime sul prezzo del giorno e aggiornano le statistiche sulle previsioni corrette di quello precedente;
  - gli agenti sono raccolti in una apposita "lista": l'ordine con cui sono registrati in tale lista viene permutata in maniera casuale in modo da garantire agli agenti ordinamenti diversi nell'immissione delle proposte;
  - fase di preapertura: immissione delle proposte di negoziazione degli agenti.
- tra t_1 e fino $t_{NumberAgent-1}$
  - fase di negoziazione continua;
- $t_{NumberAgent}$
  - viene determinato il prezzo medio della giornata;
  - gli agenti procedono al calcolo della loro ricchezza;

5.5 Gli agenti del modello

La popolazione di agenti, modificabile dall'utilizzatore della simulazione, ricrea parte della varietà delle tipologie di operatori che agiscono sui mercati finanziari. Accanto agli operatori completamente *"noise"*[14], quelli che operano casualmente, abbiamo una molteplicità di classi di agenti[15] che esprimono eterogenei schemi di comportamento. Oltre a quelli dotati di un elevato grado di capacità cognitiva (BPCT), capaci di elaborare proprie regole di azione sulla base di capacità di elaborare aspettative, avremo anche una numerosità variabile di operatori dotati di modelli comportamentali e strumenti facilmente individuabili sul mercato: dagli utilizzatori dell'analisi

[14] "Noise trading is essential to the existence of liquid market... Noise makes financial market possible, but also them imperfect."(Black, 1986) .

[15] Il termine classi qui utilizzato anticipa un concetto che sarà oggetto del prossimo paragrafo, legato all'analisi degli aspetti tecnici della simulazione.

tecnica, quali lo *StopLossAgent* e per certi versi il *marketImitatingAgent*e il *LocallyImitatingAgent* che imitano comportamenti globali o locali, a quelli che sono dotati di regole di previsione econometriche basate sulle RNA, la coppia formata da *ForecastingAgent* e *ANNForecastAppAgent*, capaci di produrre fenomeni di autorealizzazione delle aspettative.

Prima di entrare nello specifico dei vari tipi di operatori che popolano la simulazione occorre sottolineare alcune ipotesi che sono alla base del modello e che sono necessarie per la comprensione dei risultati ottenuti. Innanzitutto la ricchezza viene calcolata sommando la liquidità al valore delle azioni, sulla base del prezzo medio della seduta. Inizialmente, le dotazioni di liquidità e di azioni a disposizione degli agenti sono nulle, tuttavia agli agenti è concessa la facoltà di effettuare operazioni allo scoperto senza alcuna limitazione. Si prescinde dalla valutazione sulla possibilità di prendere a prestito o di avere dotazione infinita di risorse; l'aspetto è irrilevante allo stato attuale del modello.

Anticipando parte del contenuto del prossimo paragrafo si rende necessaria una ulteriore precisazione di carattere tecnico: tutti i parametri dei vari agenti sono variabili e potenzialmente modificabili dall'utilizzatore attraverso le opportune *probe* attivabili attraverso l'interfaccia grafica del *modelSwarm*.

L'agente base del modello è costituito dall'agente casuale (*randomAgent*) che svolge una funzione paragonabile a quella dell'operatore *noise* nei mercati organizzati. Questo tipo di agente non ha capacità cognitive: l'unica informazione di cui necessita per operare è costituita dall'ultimo prezzo eseguito. La decisione di acquistare o vendere viene presa in maniera casuale (con probabilità pari a 0.5) e la fissazione del limite di prezzo dell'ordine avviene moltiplicando l'ultimo prezzo eseguito per un coefficiente, estratto casualmente tra un range di valori modificabili dall'utilizzatore compresi tra *(minCorrectingCoeff + asymmetricRange) e (maxCorrectingCoeff + asymmetricRange)*[16].

Lo schema iniziale del modello, caratterizzato da un solo tipo di agente, quello casuale, si è progressivamente arricchito con le seguenti tipologie di agenti, tutte derivazioni ereditarie della classe base *BasicSumAgent*:

- *MarketImitatingAgent*, gli agenti che imitano il mercato: sono agenti che decidono di comperare con probabilità *asymmetricBuySellProb*[17]

[16] Dove *minCorrectingCoefficient e maxCorrectingCoefficient* sono rispettivamente il valore minimo e quello massimo del coefficiente usato per fissare il prezzo di una proposta di negoziazione di acquisto o di vendita. Gli agenti stabiliscono il loro prezzo limite moltiplicando l'ultimo prezzo eseguito per un coefficiente casuale compreso tra questi due valori. Questa correzione ha la funzione di introdurre un maggior grado di casualità nel comportamento degli agenti. Per consentire all'agente di adottare un comportamento asimmetrico è possibile aggiungere alla correzione precedente un ulteriore coefficiente espresso da *asymmetricRange*.

[17] In caso di adozione di modelli imitativi, *asymmetricBuySellProb* rappresenta la

se il prezzo medio è salito rispetto al periodo precedente, oppure di vendere con la stessa probabilità se il prezzo medio è sceso; la fissazione del limite di prezzo è identica a quella dell'agente casuale;

- *LocallyImitatingAgent*, agenti che imitano localmente: sono agenti che decidono quale operazione compiere a seconda del comportamento della maggioranza degli ultimi *localHistoryLength*[18] agenti; se a prevalere sono le proposte in acquisto (vendita), l'agente, con probabilità *asymmetricBuySellProb* formulerà una proposta in acquisto (vendita); la fissazione del limite di prezzo è identica a quella dell'agente casuale;

- *StopLossAgent*, agenti che applicano la tecnica degli stop loss[19]: sono operatori che possono agire con due modalità possibili:

  – senza memoria del passato, prescindendo dalla posizione speculativa dell'agente; in questo caso se l'ultimo prezzo eseguito nel giorno t è cresciuto (diminuito) rispetto al prezzo medio del periodo $t - stopLossInterval$[20] ad un tasso maggiore (o uguale) del parametro *maxLossRate*[21] allora l'agente compra (vende) al prezzo corrente;

  – con memoria del passato, tenendo conto della posizione dell'agente, lunga o corta sull'attività finanziaria; in questo caso, lo schema di azione descritto in assenza di memoria tiene conto della posizione dell'agente che comprerà (o venderà) se ha una posizione *short* (o *long*);

  Il comportamento dell'agente è uguale al *randomAgent* se il prezzo corrente non supera il livello fissato di *stop loss*.

- *ForecastingAgent*, agenti capaci di formulare previsioni, sotto forma di numero indice, sulla base di una struttura cognitiva costituita da una RNA; questo tipo di agente si limita a fornire previsioni sul prezzo dell'attività n giorni avanti *nAheadForecasting*[22] rispetto all'ultima media

probabilità di comprare o vendere in accordo con la strategia proposta. In assenza di tali comportamenti imitativi, come nel caso di agenti casuali, la variabile ha valore pari a 0.5.

[18] Il parametro *localHistoryLength* rappresenta la lunghezza del vettore che registra le azioni degli agenti.

[19] Strumento tipico degli operatori che compiono i loro investimenti secondo i metodi dell'analisi tecnica: impone la chiusura della posizione qualora sia stato raggiunto un livello prefissato di perdita massima, oltre il quale l'operatore non è disposto a scendere.

[20] Indica la lunghezza del periodo, orizzonte temporale, preso in considerazione nella applicazione della strategia di *stop loss*.

[21] Parametro che stabilisce il tasso di perdita oltre il quale l'agente deve attuare la strategia di stop loss.

[22] Il parametro *nAheadForecasting* indica il numero di giorni avanti a cui si riferisce la previsione della RNA; l'*output* della rete è la stima del rapporto tra il prezzo al tempo *nAheadForecasting* e l'ultimo prezzo medio disponibile.

giornaliera, senza assumere posizioni sul mercato; La RNA dell'agente ha la seguente struttura:

- *dataWindowLength*[23] nodi input; la rete in ingresso riceve gli indici di variazione dei prezzi medi giornalieri di ogni giorno rispetto al giorno *t-nAheadForecasting*;
- i nodi hidden sono pari a *dataWindowLength/2*;
- ha un solo nodo *output* che restituisce l'indice di previsione del prezzo per *nAheadForecasting* giorni avanti;

Il comportamento dell'agente è simile a quello delle varie istituzioni finanziarie che con la loro attività di informazione finanziaria, forniscono previsioni sull'andamento del mercato.

- *ANNForecastAppAgent*, agenti che seguono le indicazione della previsione basata sulla RNA: questi agenti operano in ogni seduta con probabilità *aNNForecastAppAgentActDailyProb*[24]; acquistano o vendono sulla base valore dell'indice di previsione fornito dall'agente neurale dopo un opportuno controllo sulla coerenza delle previsioni ricevute: acquistano (con probabilità pari a *asymmetricBuySellProb*) se l'indice è maggiore di *(1+ aNNInactivityRange)*[25], vendono se minore di *(1-aNNInactivityRange)*;

- *BPCTAgentA*: agenti BPCT del tipo A, dotati di una complessa struttura cognitiva, realizzata secondo la metodologia dei *Cross-Target (o CT)*[26] sviluppata in Terna (1994). Questi agenti sviluppano una coerenza interna tra l'azione di acquisto/vendita e le congetture sugli effetti relativi alla liquidità e alla quantità di azioni; possono essere considerati, in senso lato, agenti che applicano aspettative razionali. L'ipotesi sottostante la definizione di questo tipo di agenti è che un

[23] Indica il numero di dati utilizzati in input dalla RNA; tali dati sono espressi come indici calcolati rapportando il prezzo medio del giorno t con il prezzo medio del giorno *t-nAheadForecasting*.

[24] Rappresenta la probabilità di agire giornalmente sul mercato. Questo parametro è introdotto sulla base della considerazione che gli agenti che operano con previsioni non modificano il proprio comportamento continuamente.

[25] *aNNInactivityRange* è il range di variazione del prezzo previsto entro il quale l'agente rimane inattivo; se la previsione neurale è compresa tra *(1- aNNInactivityRange)* e *(1+ aNNInactivityRange)* l'agente non opera sul mercato.

[26] La definizione *Cross-Target* deriva dalla tecnica utilizzata per calcolare i *target* necessari per l'apprendimento della RNA, che rappresenta la struttura cognitiva dell'agente. La distinzione di *output* dal lato delle azioni e dal lato degli effetti porta alla necessità di incrociare i relativi *target* sui quali si forma l'apprendimento della rete. Questo incrocio, attraverso l'algoritmo della *backpropagation (BP)*, determina una serie di aggiustamenti nei pesi tra i vari strati della rete la quale sviluppa, in questo modo, la coerenza tra effetti ed azioni compiute dall'agente. Rappresenta un algoritmo al tempo stesso di apprendimento e di azione.

operatore che opera in un ambiente economico deve sviluppare ed adattare la propria capacità di valutare, in maniera coerente, quali siano le azioni necessarie per conseguire un determinato risultato e quali siano gli effetti di tali azioni. Oltre alla coerenza interna, appena descritta, gli agenti possono sviluppare altre caratteristiche, come la capacità di agire seguendo "Proposte esterne" (EP) o la capacità di valutare gli effetti sulla base di "Obiettivi Esterni" (EO) provenienti dall'ambiente e quindi anche da altri agenti. In questo modo gli schemi di comportamento dell'agente vengono sviluppati in maniera endogena come risposta anche a condizionamenti esogeni, grazie all'uso della tecnica dei CT; il comportamento non viene cioè definito a priori, ma realizzato *ex post*.

La struttura di questi agenti si basa su di una RNA così costituita:

- in *input* vi sono sette nodi: le medie dei prezzi dal giorno t-5 al giorno t-1, la liquidità dell'agente al termine del giorno t-1, le azioni dell'agente al termine del giorno t-1;
- sono presenti sei nodi *hidden*;
- in *output* ha liquidità e quantità di azioni sul lato degli effetti e la decisione di acquisto/vendita sul lato delle azioni.

Gli agenti con le loro azioni possono perseguire due differenti tipi di obiettivi esterni (EO):

1. accrescere la liquidità;
2. accrescere la quantità di azioni;

- *BPCTAgentB*, agenti BPCT del tipo B sono realizzati con la metodologia dei *cross-target* e costituiscono un'evoluzione dei più semplici agenti cognitivi di tipo A; La rete neurale che rappresenta tali agenti ha la seguente struttura:
  - otto nodi di *input*: i prezzi medi dal giorno t-5 al giorno t-1, la liquidità dell'agente al termine del giorno t-1, il numero di titoli dell'agente al termine del giorno t-1 e la previsione, formulata dall'agente neurale;
  - sei nodi *hidden*;
  - quattro *output* sul lato degli effetti: liquidità, quantità di azioni, ricchezza dell'agente valutata al prezzo di chiusura e ricchezza dell'agente valutata sulla base del prezzo previsto dall'agente

neurale; un *output* dal lato delle azioni: la decisione di acquisto/vendita[27].

La quantità acquistata o venduta è determinata arrotondando per eccesso il valore numero reale che esprime l'azione al valore intero immediatamente superiore.

Anche questo tipo di agenti cognitvi, come quelli del tipo A, persegue l'obiettivo di sviluppare una coerenza interna tra l'azione di acquisto/vendita e le congetture sugli effetti di tali azioni. L'agente cognitivo dotato di una struttura CT, sviluppa grazie all'apprendimento, non solo la capacità di valutare in modo coerente le azioni da compiere per conseguire un determinato risultato, ma anche di elaborare le conseguenze delle azioni effettivamente realizzate.

La rete avrà *target* per gli effetti e *target* per le azioni: i primi -gli effetti reali dell'azione compiuta- devono essere costruiti in coerenza con l'*output* della rete relativo all'azione (in questo caso un acquisto o una vendita di titoli), in modo da sviluppare la capacità dell'agente di stimare gli effetti dell'azione che sta decidendo; il *target* per l'azione -il comportamento necessario affinché ci sia corrispondenza con gli effetti congetturati- viene costruito in coerenza sulla base degli *output* della rete legati agli effetti, in modo da sviluppare la capacità dell'agente di decidere una azione che produca i risultati previsti.

I *target* per gli effetti sono determinati attraverso le seguenti regole:

- *target* della liquidità (E'1): è uguale alla liquidità del giorno precedente più la variazione di liquidità comportata dalla decisione di acquisto/vendita presa dall'agente nel giorno corrente;
- *target* della quantità di azioni (E'2): pari alla quantità di azioni del giorno precedente più la variazione di titoli comportata alla decisione di acquisto/vendita attuata nel giorno corrente;
- *target* della ricchezza[28] (E'3): uguale alla liquidità dell'agente più la quantità di azioni detenute moltiplicata per l'ultimo prezzo della giornata;

[27] L'azione dell'agente è rappresentata da un numero reale compreso nell'intervallo tra $\pm maxOrderQuantity$.

maxOrderQuantity indica il quantitativo massimo di acquisto o di vendita per ogni ordine immesso dall'agente. Gli agenti scelgono un quantitativo compreso tra 1 e *maxOrderQuantity*. Da un punto di vista tecnico, come accennato in precedenza, l'immissione nel book dell'ordine per un quantitativo pari a n (con $1 \leq n \leq maxOrderQuantity$) non viene realizzato in una unica soluzione, ma avviene tramite n ordini unitari: se il valore è positivo, l'agente decide di acquistare azioni, se, invece, è negativo vende. Se il valore ottenuto è pari a zero l'agente non opera sul mercato

[28] Valutata al prezzo di chiusura.

– *target* della ricchezza valutata al prezzo previsto (E'4): pari alla liquidità disponibile sommata alla quantità di azioni moltiplicata per il prezzo di chiusura al momento della previsione e per l'indice della previsione.

Per l'identificazione del *target* dell'azione (A'1), invece, è necessaria la determinazione degli adeguamenti definiti come:

– correzione per la liquidità: pari a CL = - (E1 - E'1) / PMO[29]; implica che, se la liquidità dell'agente è inferiore a quanto previsto, l'agente dovrà diminuire l'acquisto di azioni o aumentare le vendite;

– correzione per la quantità di azioni: uguale a CA = E2 - E'2; comporta che, se la quantità di azioni realmente posseduta dall'agente è inferiore alla propria congettura, l'operatore dovrà intervenire sul mercato acquistando titoli;

– correzione per la ricchezza valutata al prezzo di chiusura: pari a CRPC = (E3 - E'3) / (PC-PMO)[30]; viene valutato se la ricchezza realmente posseduta è inferiore a quella congetturata (*output* della rete o EO) e in questo caso l'agente dovrà aumentare l'acquisto (o diminuire la vendita) di titoli se il loro prezzo di chiusura è maggiore del prezzo medio operativo; se, invece, il prezzo di chiusura è inferiore al prezzo medio, l'agente deve incrementare le vendite (o diminuire l'acquisto) di azioni;

– correzione per la ricchezza valutata al prezzo previsto: ottenibile da CRPP = (E4 - E'4) / (PP-PMO)[31] ed applicabile se la ricchezza (valutata al prezzo previsto) realmente posseduta è inferiore a quella congetturata (*output* della rete o EO); in questo caso l'agente deve aumentare l'acquisto (o diminuire la vendita) di azioni se il prezzo previsto è maggiore del prezzo medio; se, invece, il prezzo previsto è inferiore al prezzo medio operativo, l'agente deve incrementare le vendite (o diminuire l'acquisto) di azioni.

Tra questi adeguamenti, la correzione più elevata in valore assoluto deve essere sommata algebricamente all'*output* A1 per poter procedere alla determinazione del *target* dell'azione.

Come nel caso degli agenti cognitivi di tipo A, anche quelli di tipo B possono perseguire due differenti tipi di obiettivi esterni (EO):

– accrescere la ricchezza valutata al prezzo di chiusura;

[29]Dove PMO è il Prezzo Medio Operativo, pari alla media dei prezzi conclusi dall'agente.
[30]CRPC = (EO - E'3) / (PC-PMO) in caso di presenza dell'obiettivo esterno.
[31]CRPP = (EO - E'4) / (PP-PMO) in caso di presenza dell'obiettivo esterno.

– accrescere la ricchezza valutata al prezzo previsto.

Gli EO dei due effetti ricchezza vengono calcolati con la somma tra il valore dell'effetto del giorno stesso, determinato dallo sviluppo dei *cross-target*, ed un ammontare fisso pari a Δ (espresso in unità di ricchezza).

5.6 Aspetti tecnici della realizzazione e del funzionamento di SUM

Il modello, come messo in luce in precedenza, è basato sul linguaggio Swarm nella sua versione Objective-C. La potenza e la flessibilità dello strumento sviluppato dal Santa Fe Institute, unitamente alle proprietà della programmazione ad oggetti, hanno reso possibile la realizzazione del modello, superando tutti gli aspetti critici della applicazione informatica: dalla gestione della memoria alla interfaccia grafica, dalla gestione del tempo alla creazione di agenti-oggetti dalle relative classi, grazie all'ereditarietà.

Riprendendo i concetti esposti nei capitoli precedenti è possibile avere un chiaro schema strutturale dell'applicazione.

In analogia con le applicazioni Swarm illustrate in precedenza, anche SUM ha come oggetti gerarchicamente più elevati l'*observerSwarm*, che gestisce i risultati della simulazione, e il *modelSwarm*, che rappresenta il modello.

Prescindendo dalle competenze in termini di funzioni informatiche svolte dalle due classi, l'interfaccia grafica dell'*observerSwarm* consente di modificare variabili relative al funzionamento di questo oggetto che si occupa della gestione e dell'analisi dei dati del modello.

Opportune *probe* dell'oggetto consentono di variare parametri quali:

- la frequenza con la quale vengono aggiornati gli oggetti che compongono l'interfaccia grafica (*displayFrequency*);
- il numero di giorni simulati al raggiungimento del quale l'esecuzione viene interrotta (*stopAtEpochNumber*);
- tutta una serie di variabili funzionali utilizzate per l'analisi, il salvataggio e la gestione dei dati prodotti dal modello e dai vari agenti che ne fanno parte.

Anche il *modelSwarm* dispone di opportune *probe* che consentono, attraverso la specifica interfaccia grafica, la gestione di importanti parametri del modello quali:

- il numero di agenti di ogni tipo (casuali, *stop loss*, imitatori, etc.) che compongono la popolazione complessiva del modello (*agentNumber*);

- una serie di parametri generali che caratterizzano tutti gli agenti della simulazione come: *asymmetricBuySellProb, minCorrectingCoefficient, maxCorrectingCoefficient, asymmetricRange, agentProbToActBeforeOpening*[32]*, floorP*[33]*, agentProbToActBelowFloorPrice*[34] *e maxOrderQuantity*

- tutte le variabili specifiche che caratterizzano le varie tipologie di agenti inserite nel modello.

Per essere rigorosi, la creazione di agenti più complessi quali il *forecastingAgent* e il *BPCTAgent* può richiedere l'inserimento di specifici parametri in appositi file di servizio[35], comunque, non indispensabili ad eccezione di quelli che contengono i valori di passaggio dalla metrica esterna di *input* e *output* alla metrica interna della RNA degli agenti cognitivi.

observerSwarm, *modelSwarm*, e gli agenti dalla simulazione sono istanze delle omonime classi Swarm che compongono la struttura gerarchica del modello. Le classi principali dell'applicazione sono le seguenti:

- *ObserverSwarm*: definisce l'observer del modello;

- *ModelSwarm*: è la classe che definisce il model della simulazione; in questa classe sono attivati i messaggi alle classi da cui vengono istanziati i diversi agenti, viene create una istanza del book e viene gestita la scansione degli eventi che costituisce il timing della simulazione;

[32] Indica la probabilità di agire in fase di preapertura.

[33] Rappresenta il valore del *floor* per il prezzo.

[34] Fornisce la probabilità di comprare quando il prezzo scende al di sotto del *floor*.

[35] Segue un sintetico elenco di tali *file* e delle relative informazioni contenute:

- *forecasting.setup*: contiene i valori dei parametri utilizzati dalla RNA che realizza l'agente neurale;

- *transf.setup*: contiene il valore k della funzione logistica

$$f(x) = \frac{1}{(1 + e^{-kx})}$$

utilizzata dalla RNA quale funzione di attivazione;

- *bp.setup*: contiene i parametri di funzionamento delle reti neurali CT utilizzate per la realizzazione degli *BPCTAgent*;

- *minmaxA.data*: determina i valori di passaggio dalla metrica esterna di *input* e *output* dell'agente cognitivo A alla metrica interna della RNA dell'agente;

- *minmaxB.data*: determina i valori di passaggio dalla metrica esterna di *input* e *output* dell'agente cognitivo B alla metrica interna della RNA dell'agente;

- *initA.val*: contiene i valori iniziali in metrica esterna di input e output dell'agente cognitivo di tipo A;

- *initB.val*: contiene i valori iniziali in metrica esterna di input e output dell'agente cognitivo di tipo B.

- *Matrix, Matrix2, MatrixMult*: sono classi che definiscono i metodi necessari per la gestione delle strutture numeriche (vettori e matrici) che contengono i dati utilizzati dal modello;

- *Book*: questa classe fornisce gli strumenti necessari a garantire il funzionamento del *book* del mercato in tutte le sue fasi e modalità; in tale classe sono contenuti i metodi che consentono l'immissione degli ordini da parte degli agenti, il loro abbinamento automatico, la registrazione (*log*) delle proposte ineseguite;

- *BasicSumAgent*: è la classe base degli agenti in SUM da cui ereditano variabili istanza e metodi (compresi quelli di *accounting*) tutti gli agenti, con la sola eccezione del *forecastingAgent*;

- *BasicSumRuleMaster*: è la classe base dei *RuleMaster* da cui ereditano variabili istanza e metodi i gestori di regole di tutte le tipologie di agenti, eccetto quelli del previsore neurale e degli agenti cognitivi.

La gestione degli agenti del modello ha richiesto la creazione di apposite istanze della classe delle "liste" di Swarm; questi oggetti, concettualmente simili a vettori, favoriscono la gestione delle allocazioni di memoria dei singoli agenti, identificati dal relativo numero indice, e curano l'invio dei messaggi. Le gestione delle funzioni di *messaging* è semplificata dall'uso di questi oggetti: per inviare un messaggio ad un gruppo di agenti sarà sufficiente inviarlo alla lista che li contiene.

Durante la fase di sviluppo del modello è stata posta particolare attenzione agli aspetti legati alla casualità di processi e variabili: sono stati creati una serie di generatori di numeri casuali indipendenti per le varie tipologie di agenti oltre ad un apposito oggetto (definito *Shuffler*) a cui è demandata la funzione di permutare gli elementi delle varie liste. Questo per garantire la massima indipendenza dalle condizioni iniziali e da eventuali correlazioni interne al modello.

La gestione dell'esecuzione della simulazione avviene grazie alle funzionalità messe a disposizione dell'utilizzatore dal "ControlPanel"; questa interfaccia grafica non solo permette l'avvio e l'interruzione, anche solo temporanea, della simulazione ma consente anche l'analisi delle strutture interne dei vari agenti attivati, permettendo di conoscere stato e dinamica delle principali variabili. Questa possibilità è garantita dal metodo *openProbeTo* che permette l'accesso ai dati dei singoli agenti delle varie liste, identificati dal relativo numero indice.

Ciò permette allo sperimentatore di analizzare gli aspetti microeconomici degli agenti rappresentati, in linea con l'impostazione *"bottom-up"* tipica del paradigma simulativo.

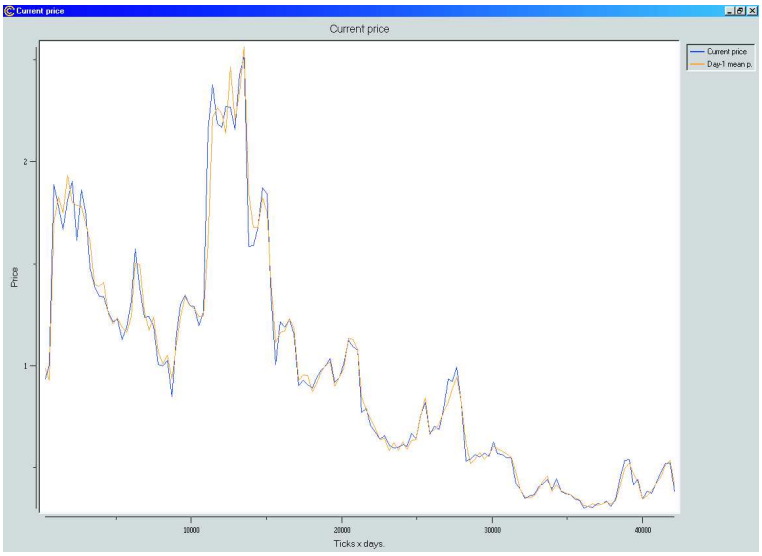

Figura 5.1: Dinamica del prezzo determinata dall'interazione di 300 agenti *random*

5.7 Risultati della simulazione

Sulla base dello schema delineato in Terna (2001), sono state eseguite una serie di simulazioni derivanti dall'applicazione di SUM con l'obiettivo di analizzare gli effetti prodotti dalla presenza e dall'interazione di diverse tipologie di agenti sulle micro e macro strutture del mercato simulato.

Anticipando le conclusioni, un importante risultato ottenuto dal modello è stato quello di generare serie di tempo crescenti e decrescenti con elevata volatilità e formazione di bolle speculative e *crash*, come conseguenza delle semplici regole del mercato. Tale comportamento si è verificato sin dalla configurazione più semplice del mercato: quella caratterizzata dalla esclusiva presenza di agenti casuali.

Questo risultato ha avuto il pregio di mostrare come comportamenti complessi possano emergere anche dall'interazione di agenti estremamente semplici, senza necessariamente ricorrere all'uso di strutture cognitive BDI. La sofisticata struttura del mercato, rappresentata dal *book*, si è mostrata capace di riprodurre in maniera endogena sequenze di prezzo dipendenti dalle azioni degli agenti secondo modalità non lineari.

Le dinamiche delle serie dei prezzi e la formazione di bolle e *crash* non sono effetto di cause esogene al modello, ma trovano la loro spiegazione nella struttura del mercato. Il modello è profondamente influenzato dalle regole di funzionamento del *book* elettronico, che non solo condiziona i comportamenti individuali degli agenti, ma anche i meccanismi di interazione tra gli stessi.

Dall'analisi del registro (*log*) del *book* emerge, infatti, che gli incrementi dalla volatilità dei prezzi sono anticipati da squilibri tra le proposte di segno opposto espresse dagli operatori. Squilibri con forti quantità in acquisto e scarse in vendita provocheranno la comparsa di bolle speculative; forte pressione in lettera e scarsa reattività del danaro, invece, anticiperà le fasi opposte caratterizzate da una rapida caduta dei prezzi (*crash*).

Questi squilibri trovano conferma dalla presenza di lunghe sequenze di ordini su uno stesso lato del mercato (acquisto o vendita) tanto più marcate quanto maggiore è il quantitativo massimo negoziabile dagli agenti. Se l'ammontare di titoli su cui può operare un singolo agente è contenuto, il mercato mostra una sostanziale stabilità, e a prevalere sarà il "rumore bianco". Se, invece, agli agenti è concesso di negoziare quantitativi elevati, allora la probabilità di squilibri tra i due lati del *book* e di successive accelerazioni nei prezzi sarà maggiore.

Una ulteriore verifica all'ipotesi arriva dalla analisi della dinamica del rapporto tra ordini di acquisto e ordini di vendita non eseguiti: durante le fasi di veloce accelerazione nei prezzi con tendenza al rialzo, questo indicatore si mantiene su livelli estremamente elevati, raggiungendo i propri massimi in corrispondenza con quelli del mercato. L'andamento opposto registrato dall'indicatore nelle fasi veloce ribasso conferma l'assunto che sia la sistematica assenza di una controparte per gli scambi a far aumentare la pressione su uno dei due lati del *book*.

Analisi statistiche hanno evidenziato la capacità della componente legata agli ordini ineseguiti nello spiegare gli incrementi di variabilità nei prezzi.

Con l'introduzione di molteplici categorie di operatori con diversi gradi di capacità cognitiva, i risultati raggiunti sono stati confermati e il modello SUM si è mostrato capace di ricreare altri fenomeni simili a quelli riscontrati nei mercati reali.

L'inserimento di agenti specializzati nell'imitare i comportamenti globali o locali, anche in numero limitato, ha determinato forti incrementi nella volatilità del mercato. Gli agenti che adottano strategie imitative del mercato riescono a limitare il livello di perdite nei periodi di *crash* successivi alle bolle speculative, ma non riescono a raggiungere i livelli di guadagno raggiunti dagli agenti casuali.

Risultati decisamente migliori per gli imitatori locali che riescono a contenere le perdite nelle fasi di sgonfiamento delle bolle e registrano un livello medio di ricchezza maggiore degli agenti casuali. A differenza degli imitatori del mercato, questo tipo di imitatori sembra maggiormente capace di sfruttare le opportunità di guadagno offerte dal *trend* crescente. A livello aggregato, il mercato con l'introduzione di agenti che imitano localmente presenta una dinamica di prezzo che manifesta fenomeni speculativi decisamente più contenuti.

L'adozione di una strategia fortemente difensiva penalizza il livello di ric-

chezza degli agenti *stop loss*, sempre prossima allo zero e per nulla influenzata dalle bolle e dai *crash* che si verificano nel mercato.

Il modello progressivamente esteso con l'esplorazione di molteplici configurazioni[36], caratterizzate da gradi di complessità degli agenti crescenti, ha, inoltre, mostrato l'emergere di processi di autorealizzazione delle aspettative.

Oltre ad analizzare la dinamica dei prezzi, il modello rappresenta un valido strumento per consentire la valutazione del grado di prevedibilità del mercato artificiale grazie all'analisi delle stime prodotte dal previsore neurale.

Gli agenti *ANNForecastAppAgent* decidono le loro azioni in maniera coerente con le stime sull'andamento dei prezzi elaborate dall'agente previsore neurale. Questo agente, che non opera in maniera diretta sul mercato, sviluppa delle previsioni sulle dinamiche future dei corsi, registrando sistematicamente la percentuale di previsioni corrette sul totale delle stime formulate. La qualità di queste stime dipende dal grado di prevedibilità del mercato che sarà tanto maggiore quanto più elevato sarà il numero di agenti che applica queste previsioni. Conseguenza di ciò sarà un marcato incremento del livello di prevedibilità del mercato stesso e della percentuale di successo delle stime. La retroazione dell'effetto sembra rientrare pienamente nell'ambito delle previsioni autorealizzanti.

Questo fenomeno ha influenze ovviamente sulla ricchezza degli agenti che seguono questo schema. La loro ricchezza presenta un maggiore livello di stabilità rispetto agli operatori casuali.

L'introduzione anche dell'ultimo tipo di agenti, quelli caratterizzati da strutture cognitive complesse, mantiene inalterate le conclusioni raggiunte. Il comportamento di questi agenti risulta profondamente influenzato dalla forza dell'obiettivo esterno "aumentare la ricchezza valutata sul prezzo previsto", fattore capace di produrre schemi di comportamento, e risultati connessi, profondamente differenti da una configurazione all'altra. La capacità cognitiva dell'agente ha dato risultati solo per opportune condizioni di questo parametro; in questi casi, gli operatori BPCT hanno saputo approfittare delle fasi di volatilità dei prezzi per incrementare sensibilmente il proprio livello di ricchezza, anticipando i movimenti del mercato.

A livello aggregato, queste fasi, ancora caratterizzate da fenomeni speculativi, hanno visto aumentare in maniera sensibile la capacità revisionale degli operatori BPCT, che, assieme agli *ANNForecastAppAgent*, ottengono le migliori prestazioni.

In altre ciò non si è verificato.

L'analisi statistica compiuta per individuare gli schemi d'azione degli agenti BPCT non ha fornito risultati significativi: lo studio ha evidenziato

[36] Agendo non solo sulla popolazione degli agenti, ma anche sulle variabili ed i parametri della simulazione.

Figura 5.2: Il logo di SUMWEB

l'emergere di una complessa struttura comportamentale non riconducibile a fattori ben individuabili ed isolabili.

5.8 Gli sviluppi di SUM: l'evoluzione SUMWeb

I significati risultati raggiunti dal modello non hanno rappresentato un punto di arrivo per il progetto, tuttora in piena fase di sviluppo. Volontà dell'autore è quella di aumentarne progressivamente il grado di realismo, eliminando parte delle semplificazioni introdotte, estendendo meccanismi e strutture del mercato e ricreando gran parte della varietà dei comportamenti attuati dagli operatori "reali".

Il primo passo di un simile piano di sviluppo ha comportato l'"apertura" del modello verso il mondo esterno, dando la possibilità ad agenti "umani" di operare in competizione con quelli artificiali. Questa evoluzione, oggetto di presente elaborato, ha dato vita ad una particolare versione dell'applicazione, chiamata SUMWeb (SUM Web economic behaviour) a causa dell'utilizzo dei protocolli del *world wide web* per la gestione dei processi.

Passi successivi del modello saranno rappresentati dall'estensione del numero di attività trattate, moltiplicando il numero dei *book* e di tutte le strutture del mercato, la creazione di un indice rappresentativo dell'andamento complessivo di questi titoli e dalla introduzione di un mercato a scadenza, in cui negoziare un "future" sull'indice precedentemente determinato. Particolarmente interessante in questa è la definizione di una nuova classe di agenti, provvisoriamente detta "agente-arbitraggista", capace di valutare eventuali scostamenti tra valore a termine e a pronti dei titoli e di agire in modo da riportare l'equilibrio sui due mercati. Questo comportamento conferma ancora una volta la volontà di escludere meccanismi di controllo e coordinamento a livello aggregato delle quotazioni.

Ulteriore sviluppo del modello sarà rappresentato dall'introduzione dell'informazione negli schemi di comportamento degli agenti. A questo proposito, l'ipotesi iniziale prevede la creazione di una specifica classe di operatori

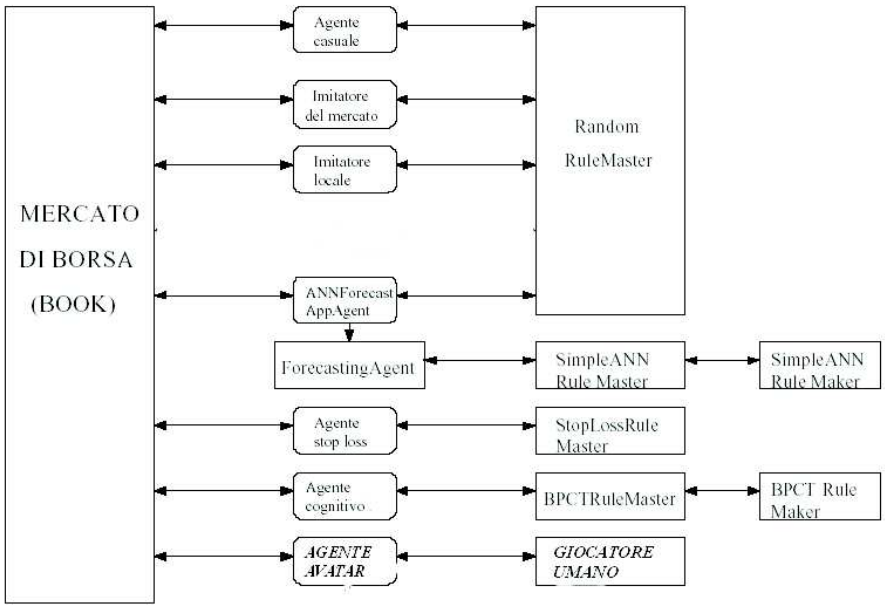

Figura 5.3: Lo schema ERA applicato a SUMWEB

in grado di reagire ad eventi generati da un apposito "generatore di notizie", operando in modo da riflettere il nuovo set di dati nei prezzi delle attività.

Ultimo passo previsto per il modello dovrebbe essere rappresentato dall'integrazione con un apposito *data base*, sviluppato su piattaforma mySQL[37], in grado di gestire la mole di informazioni prodotta dall'esecuzione ripetuta e prolungata della simulazione.

5.9 SUMWeb

Il nome SUMWeb, come anticipato in precedenza, deriva dall'uso dei protocolli legati al *web* per la gestione della simulazione, e del relativo flusso informativo, estesa alla possibilità di azione da parte di agenti "umani".

SUMWeb riprende schema e caratteristiche originali del modello SUM e grazie ad una nuova classe di agenti, *AvatarAgent*, è in grado di far partecipare operatori reali al processo di formazione del prezzo e di perfezionamento degli scambi, in condizioni di elevato realismo.

L'estensione del modello verso l'esterno, compiuta con la versione SUMWeb, ha posto gli sviluppatori di fronte ad una serie di problematiche connesse da un lato con lo sviluppo e la modifica del codice originale di SUM,

[37] A questo proposito si veda il paragrafo in appendice.

basato sul linguaggio Swarm Objective-C, e dall'altro con la creazione di una interfaccia grafica capace di garantire la trasmissione bidirezionale di informazioni (nello specifico ordini ed eseguiti) tra il mondo simulato e l'esterno.

Il modello seguito per la creazione dell'interfaccia è stato rappresentato, con le opportune semplificazioni, dall'insieme di strumenti grafici messi a disposizione dai principali operatori bancari attivi nel servizio di *trading on line*.

5.9.1 Il codice Swarm

Lo sviluppo e la modifica del codice[38] dell'ultima versione di SUM ha inizialmente comportato la necessità di individuare all'interno del listato le variabili e i metodi su cui intervenire per proiettare all'esterno i risultati ottenuti dalla simulazione.

Il livello minimo di informazioni che si è reso necessario distribuire ad un investitore umano[39] che operi all'interno del mercato simulato è rappresentato da:

- tempo della simulazione, identificato alla variabile *dayNumber* definita in *ModelSwarm.m*;
- ultimo prezzo eseguito, identificato nella variabile *executedPrice* definita in *Book.m*;
- proposte contenute nei vari livelli del *book*, rappresentate in forma aggregata per ogni prezzo con indicazione della quantità complessiva. Per la costruzione di questa modalità di visualizzazione, realizzata graficamente all'interno della pagina *web*, si è resa necessaria la gestione delle due variabili, *buyOrderNumber* e *sellOrderNumber*, e dei relativi metodi[40], possono essere così rappresentati:

```
- printNRowsFileB: (int) nr {
  int i;
    FILE * pFile;

    if(nr==0){printf("Zero rows in Code=

    %1d matrix\n",code);return self;}
```

[38] Allegato in appendice.

[39] Prescindendo da quale metodo o strumento gestionale applichi nella propria attività di *trading*.

[40] I metodi, *printNRowsFileB* e *printNRowsFileS* sono definiti e resi eseguibili nei file *matrix2.h* e *matrix2.m*. I metodi, inviati rispettivamente agli oggetti *buyOrderStorehouse* e *sellOrderStorehouse* definiti e implementati nei file book.h e book.m

```
    pFile = fopen ("Buy.dat","w");
    if (pFile == NULL) {
    perror("cannot open output file");
    exit(1);
    }

   for (i=0;i<nr;i++)
   {
   fprintf(pFile,"
   %7.4f <br>\n",m[i*cols]);
   }
  fclose(pFile);
  return self;
}

- printNRowsFileS: (int) nr {
  int i;
  FILE * pFile;

  if(nr==0){printf("Zero rows in Code=
  %1d matrix\n",code);return self;}

    pFile = fopen ("Sell.dat","w");
    if (pFile == NULL) {
    perror("cannot open output file");
    exit(1);
    }

   for (i=0;i<nr;i++){
    fprintf(pFile,"
    %7.4f <br>\n ",m[i*cols]);
}
  fclose(pFile);
  return self;
}
```

attraverso le istruzioni presenti nel file book.m:

```
[buyOrderStorehouse printNRowsFileB: buyOrderNumber];
```

per le proposte in acquisto e

```
[sellOrderStorehouse printNRowsFileS: sellOrderNumber];
```

per quelle in vendita.

Queste modifiche hanno richiesto una notevole attenzione nell'identificazione di istruzioni in Objective-C per la scrittura e lettura di informazioni[41] su *file* (con estensione .dat) e per la gestione di questo tipo di file. La necessità di scrittura e lettura spesso si sono incrociate con quelle legate al controllo della presenza o meno del *file* di provenienza o destinazione delle informazioni e al controllo dei tipi di dati contenuti. In alcuni casi si è reso necessario all'interno dello stesso blocco di istruzioni creare e poi cancellare lo stesso *file*; in altri, i nuovi dati sono stati semplicemente inseriti in coda ai precedenti con l'opzione *append*, propria dei metodi del C per la gestione dei *file*.

Le operazioni e le istruzioni descritte non sono solo state necessarie per il controllo e la visualizzazione dell'*output* prodotto da SUM, ma hanno caratterizzato anche la successiva, in ordine cronologico, fase di inserimento dell'*input* da parte degli agenti "umani".

Per la realizzazione di un sistema di gestione dell'input si è resa necessarie la creazione di una apposita classe di agenti, quella dell'Avatar. Il dizionario riporta:

> **AVATAR o AVATARA** (dal sanscrito). Letteralmente significa discende da molto lontano ed indica una Incarnazione divina in un essere umano di un Essere altamente evoluto che non ha più la necessità di rinascere in un corpo mortale.

In ambito informatico, la parola è diventata di uso comune per indicare la possibilità data ad un oggetto di rappresentare una sorta di *alter ego* di un altro.

Nell'ambito della simulazione SUMWeb, la classe *AvatarAgent*[42] ha il compito di permettere la creazione di una numerosità di istanze, *avatarAgent*, che fungono da rappresentazione del giocatore reale all'interno della popolazione di agenti simulati.

Queste istanze ereditano le funzioni base che caratterizzano tutti gli agenti di SUM dalla classe BasicSumAgent che ne permettono la gestione a livello informatico sfruttando le proprietà dell'ereditarietà della programmazione ad oggetti.

Le istanze *Avatar*, come detto in precedenza, oltre ad essere fisicamente inseriti in una specifica lista (*avatarAgentList*) si sommano agli altri agenti della popolazione con cui possono interagire con gli stessi metodi e le stesse funzionalità.

Sono le regole di azioni che differenziano gli agenti *Avatar* dal resto della popolazione: se il comportamento degli agenti di SUM derivava da regole definite a priori o da regole autogenerate e gestite da appositi *RuleMaker* e *RuleMaster* come nel caso del *BPCTAgent*, quello degli *Avatar* deriva dalle decisioni prese dal giocatore umano di fronte al mutare dell'ambiente.

[41] Principalmente *fscanf* e *fprintf*.

[42] Definita ed implementata negli omonimi *file: AvatarAgent.h* e *AvatarAgent.m*.

Il giocatore esprime le proposte di acquisto/vendita in pre-apertura o in trattazione continua attraverso una opportuna interfaccia *web*, oggetto di trattazione nel prossimo paragrafo: questo strumento, oltre a consentire la visualizzazone delle variabili definite in precedenza, genera ad ogni giocata uno specifico *file (order.dat)* che contiene i principali parametri che caratterizzano la proposta:

- tipo di operazione: attraverso lo *switch*binario (1,-1) ogni proposta viene gestita da SUM come un acquisto (se 1) o una vendita (se -1);
- prezzo limite: il massimo (o il minimo) prezzo a cui l'operatore umano si dichiara pronto ad acquistare (vendere);
- quantità: la quantità oggetto della proposta.

La classe *AvatarAgent* fornisce alle proprie istanze gli strumenti necessari per la creazione, la scrittura e la cancellazione del *file* contenente le informazioni indicate e la successiva conversione in dati utili a SUM per la gestione della proposta inserita.

Entrato nel mondo virtuale, l'ordine pseudo reale sarà elaborato dal book di SUM che si farà carico di registrare la proposta e di verificare la presenza o meno (presente e futura) di una eventuale controparte con cui perfezionare lo scambio.

Tecnicamente, i singoli agenti *Avatar*vengono chiamati ed attivati con il messaggio di attivazione del metodo act1 passato alla lista che li contiene. L'istruzione fisicamente contenuta nel *modelSwarm* è la seguente:

```
if (avatarAgentNumber>0)[modelActions2 createActionForEach:

 avatarAgentList message: M(act1)];
```

e viene elaborata ed eseguita, ad ogni ciclo, dopo aver attivato la lista del *currentAgent* che contiene gli altri agenti con il seguente comando:

```
[modelActions2 createActionTo:
theCurrentAgent message: M(act1)];
```

La sequenza di eventi descritta è necessaria per consentire al giocatore umano di esprimere le proprie intenzioni negoziali in ogni momento della seduta, indipendentemente dall'ordine attribuito alla lista degli altri agenti stabilito dal metodo *listShuffler* presente in *modelSwarm*.

Le future versioni di SUM dovrebbero eliminare il rudimentale ricorso alla scrittura su *file* per la gestione delle informazioni ed adottare una ben più funzionale ed efficiente struttura di dati, basata sull'ambiente *database* mySQL, perfettamente integrata con l'interfaccia *web*, realizzata in PHP.

5.9.2 L'interfaccia *web*

Lo strumento attraverso il quale i giocatori umani accedono al mercato simulato è rappresentato da un interfaccia grafica, realizzata ricalcando le funzionalità di base messe a disposizione ai propri clienti dalle società di *trading on line*.

La simulazione, attivata e gestita attraverso l'architettura *client-server* messa a disposizione da *vncserver*, è in esecuzione su di un *server* RedHat del Dipartimento di Economia. La presenza del *webserver* Apache permette al giocatore umano di prendere parte alla simulazione attraverso l'apertura di una sessione di un qualunque *browser internet*.

Il giocatore umano interagisce con il mondo simulato digitando in una finestra del proprio *browser* l'indirizzo del sito:
http://eco83.econ.unito.it/sumweb
che punta alla cartella nel server in cui risiede la simulazione. Compiendo questa operazione, il giocatore si troverà di fronte una pagina scritta in codice html, nominata index.html, che ha la funzione di leggere e scrivere i dati contenuti nei *file* di servizio di SUMWeb.

La pagina è suddivisa in più cornici, ognuna delle quali rappresenta un ulteriore documento in formato PHP[43] a cui è demandata la funzione di consentire la visualizzazione o l'inserimento dei dati presenti sui vari *file* di servizio.

Anche in questa parte del lavoro si è reso necessario fare un ampio uso delle istruzioni del linguaggio di scripting per la gestione, la scrittura e la cancellazione di dati su *file*[44].

Allo stato attuale di SUMWeb, la pagina visualizza tutte le informazioni che costituiscono l'*output* di SUM:

- il giorno della simulazione: informazione contenuta nel *file* di servizio day.dat e visualizzata dalla cornice day.php;
- l'ultimo prezzo scambiato informazione contenuta nel *file* lastsingle.dat e visualizzata dalla cornice grafici.php;
- i dati relativi al *book* di SUM visualizzati nella cornice book.php ed espressione delle informazioni contenute nei *file*:
  - buy.dat per le proposte di acquisto, inserite da tutti gli operatori del mercato, rappresentate in ordine decrescente, con indicazione di:
    a) livello di prezzo delle proposte;
    b) quantità aggregata di domanda per ogni livello di prezzo;

[43] Per maggiori informazioni su questo potente e flessibile linguaggio di *scripting* si veda l'appendice.
[44] Principalmente fputs ed fgets per l'*input* e l'*output* delle informazioni.

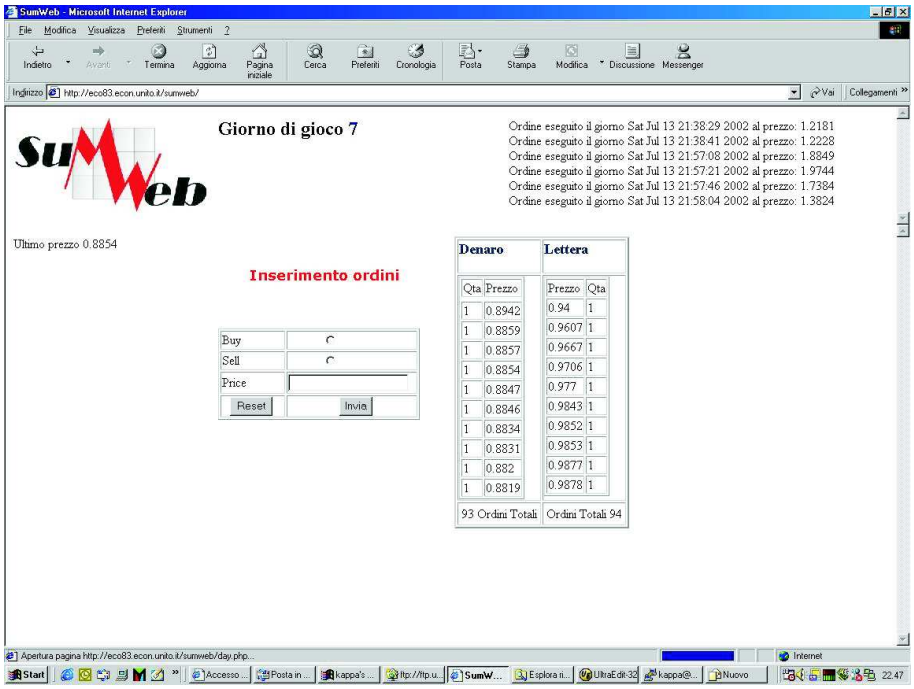

Figura 5.4: L'interfaccia *web* di SUMWEB

– sell.dat per le proposte di vendita, inserite da tutti gli operatori del mercato, rappresentate in ordine crescente, con indicazione di:

  a) livello di prezzo delle proposte;
  b) quantità aggregata di offerta per ogni livello di prezzo.

La versione attuale di SUMWeb non sfrutta ancora in maniera completa le sei cornici, pensate in funzione dei futuri sviluppi che dovrebbero portare inizialmente alla visualizzazione di grafici e statistiche di mercato (cornice grafici.php) e alla gestione di un sistema di *messaging* tra mondo simulato e giocatori (cornice messages.php). Quest'ultimo ha avrà il compito di fornire al giocatore informazioni personali, riguardanti statistiche ed informazioni sulle proprie operazioni, e collettive, principalmente legate ad eventi dell'ambiente simulato.

L'ultima, ma non meno importante, cornice della pagina è rappresentata da user.php che ha lo specifico compito di gestire le informazioni di *input* provenienti dal giocatore. La maschera riproduce i campi tipici di un'interfaccia operativa di una piattaforma di *trading on line* e consente all'utilizzatore di inserire le proprie proposte negoziali con l'indicazione di:

- tipo di operazione (acquisto/vendita);

- prezzo limite della proposta;
- quantità della proposta.

Completati i campi dell'ordine, il giocatore potrà o inviare l'ordine al mercato, premendo il tasto di controllo "invia", o cancellare i dati inseriti e azzerare i campi con il tasto "reset".

user.php utilizza le funzioni di scrittura su *file* proprie del linguaggio PHP per generare il *file* di servizio order.dat, utile all'agente Avatar della simulazione SUM per conoscere le intenzione di acquisto o di vendita del giocatore.

Il codice scritto permette un controllo preventivo delle informazioni inserite dal giocatore sia in termini di coerenza informatica del tipo di dati sia in termini di coerenza con la dinamica dei prezzi del mercato (controllando l'eventuale scostamento percentuale dall'ultimo prezzo scambiato). L'interfaccia è in grado di comunicare all'utilizzatore se l'ordine contiene informazioni difformi da quelle richieste, di segnalare eventuali eccessi di scostamento del prezzo inserito rispetto al *last* e di informare sull'inserimento corretto della proposta nel *book*.

5.10 La configurazione ottenuta e i risultati raggiunti

Realizzata l'architettura SUMWeb, grande attenzione è stata posta ad aumentare il grado di realismo, in una diversa accezione rispetto a quanto visto in precedenza, della dinamica del mercato. Se per SUM, e di conseguenza per SUMWeb, il realismo era inteso in termini di capacità di riprodurre fedelmente il reale, ora nell'ambito della sperimentazione la ricerca del realismo è volta a far sembrare come "quasi-reale" quanto percepito dall'essere umano.

La prima difficoltà incontrata è stata rappresentata dalla non coincidenza del tempo fisico del giocatore con quello della simulazione. L'eccessiva velocità del trascorrere del tempo in SUM[45] ha comportato la necessità di introdurre specifici meccanismi di rallentamento dei cicli per permettere al giocatore la corretta percezione e valutazione delle informazioni ricevute.

Reso più realistico il trascorrere del tempo, si è reso necessario confrontare il comportamento del prezzo dell'attività simulata con la dinamica dei prezzi reali. Riprendendo gli esperimenti compiuti con SUM, l'ambito di analisi è stato costituito dalla ricerca di una configurazione dei parametri del sistema in grado di produrre andamenti di prezzo non troppo difformi da quelli reali.

Per guidare il processo di affinamento dei parametri al fine di incrementare il realismo dei risultati sono state assunte come variabili dipendenti

[45] Determinato dalla successione degli eventi dello *schedule*.

Figura 5.5: Il pannello di controllo di SUMWEB

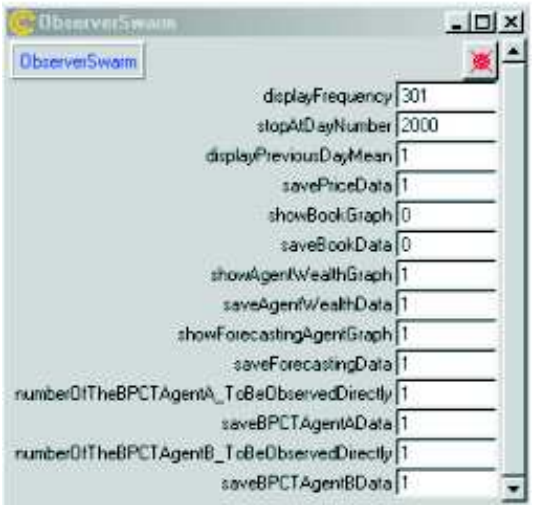

Figura 5.6: L'ObserverSwarm di SUMWEB

la volatilità e le variazioni giornaliere dei prezzi. Questa scelta ha comportato la necessità di agire sulla numerosità e sul tipo degli agenti artificiali al fine di individuare una popolazione in grado di produrre dinamiche di prezzo coerenti con quelle osservate.

Le osservazioni compiute hanno messo in luce i seguenti requisiti:

- ipotizzare popolazioni numerose per aumentare la variabilità del sistema;
- costituire la popolazione con un considerevole numero di agenti casuali per impedire eccessivi scostamenti nei prezzi;
- limitare il numero di agenti dotati di schemi di comportamento imitativi per evitare fenomeni di inerzia delle quotazioni capaci di produrre gli effetti esplosivi delle bolle.

Alla luce di queste considerazioni, le prove compiute hanno mostrato come una simulazione caratterizzata da:

- 1200 *randomAgent*;
- 30 *StopLossAgent*;
- 20 *marketImitatingAgent*;
- 15 *LocallyImitatingAgent*

sia stata in grado di fornire un comportamento della serie dei prezzi compatibile con il grado di realismo, richiesto in termini di volatilità e rendimenti giornalieri dei prezzi.

La simulazione così configurata, attiva *on line*, si è rivelata capace di restare in esecuzione per un tempo indeterminato, senza manifestare problemi o malfunzionamenti. L'esecuzione continua dei processi non ha provocato alcun decadimento di prestazioni al *sever* su cui è installata. Questi risultati hanno confermato l'ottima capacità del codice SUM, scritto utilizzando le funzioni di Swarm, nella gestione dell'allocazione della memoria dell'elaboratore.

Solo al fine di evitare un eccessivo appesantimento dei *file* di log generati dalla simulazione[46] si è reso necessario prevedere un sistema di riavvio automatico e periodico dell'applicazione. Per ottenere questo risultato si è agito sui controlli del *modelSwarm*, capaci, se opportunamente configurati, di chiudere e far ripartire SUMWeb in maniera autonoma.

[46] Creati *ex-novo* ad ogni nuova esecuzione.

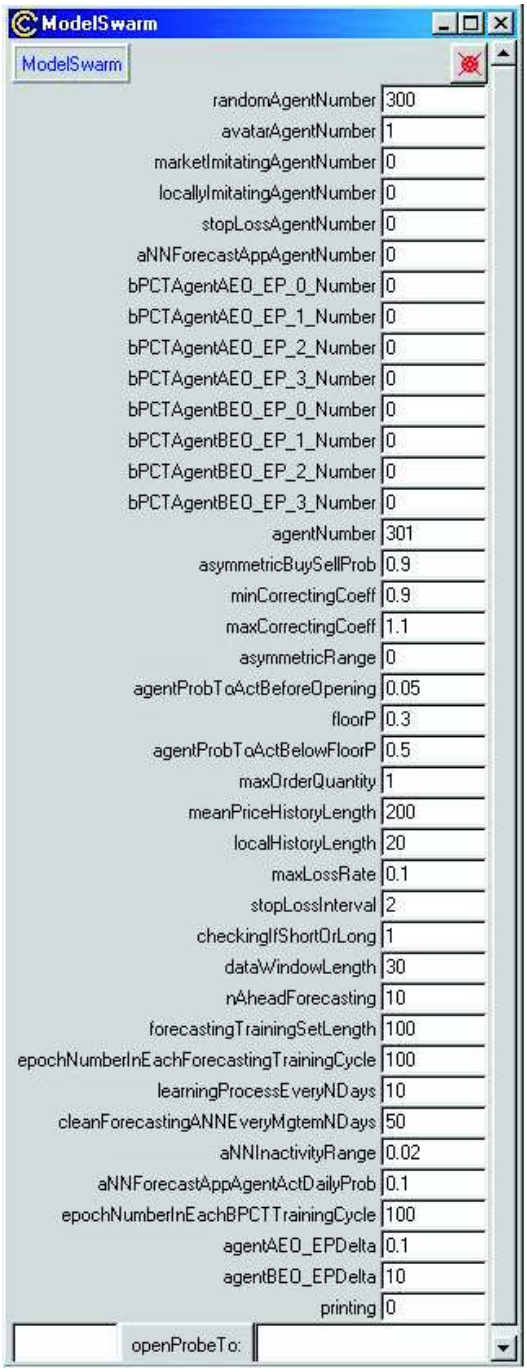

Figura 5.7: Il ModelSwarm di SUMWEB

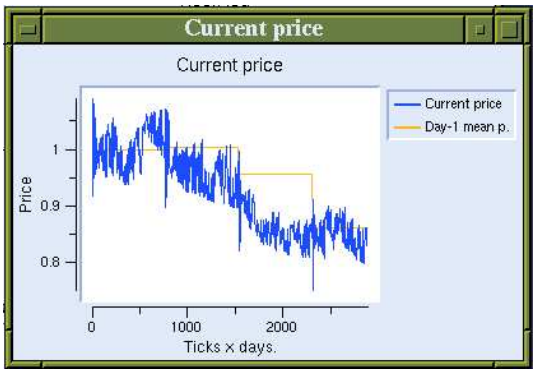

Figura 5.8: L'esecuzione della simulazione SUMWEB: la serie dei prezzi

5.11 Conclusioni

Il modello allo stato attuale rappresenta un autentico laboratorio sperimentale in grado di consentire lo studio di comportamenti economici, psicologici e sociali associati all'evoluzione dei prezzi in un mercato regolamentato, definito con le caratteristiche indicate.

La possibilità, offerta dalla scelta di utilizzare una architettura informatica aperta all'esterno, di coinvolgere una vasta comunità di giocatori, potenzialmente costituita da tutti coloro che hanno un accesso alla rete *web*, aumenta l'interesse generato da una simulazione realizzata secondo i criteri descritti. Le peculiarità dell'applicazione elencate in precedenza conferiscono all'esperimento elementi di novità rispetto a lavori analoghi di altri autori.

Gli sviluppi in campo dell'economia sperimentale apportati dal modello hanno messo in luce le differenze rispetto ad altri lavori sviluppati. In molti casi, i primi esperimenti tentati dagli studiosi hanno creato ambienti di azione artificiali, spesso lontani da attinenze con aspetti dalla realtà. SUMWeb, invece, riproducendo fedelmente l'interfaccia grafica di un qualunque *tool* di *trading on line*, pone i soggetti osservati a confrontarsi con un ambiente già noto e sostanzialmente realistico.

Dallo studio dei loro comportamenti, e dall'analisi delle interazioni derivanti, potranno essere osservati interessanti fenomeni non solo capaci di permettere lo studio dei mercati finanziari come sistemi complessi all'economia e alla finanza, ma anche di verificare aspetti di psicologia e sociologia che caratterizzano l'azione degli investitori.

La esecuzione ripetuta dell'esperimento, opportunamente dotato di strumenti in grado di effettuare la rilevazione delle variabili oggetto di osservazione potrebbe permettere di analizzare aspetti quali:

- la propensione al rischio mostrata dai singoli giocatori;

- comportamenti in linea con l'ipotesi di razionalità, o meno, degli operatori;
- comportamenti imitativi degli operatori;
- il valore dell'informazione negli schemi di azione;
- la capacità di elaborazione delle informazioni;
- la risposta a impulsi di carattere emotivo;
- i comportamenti aggregati e la possibilità di situazioni di euforia o di panico tipiche delle bolle.

Questi sono solo alcuni dei temi di analisi che potenzialmente possono essere studiati con la versione *web* di SUM. Se è vero che:

> "Mostrare, via simulazione, la plausibilità del funzionamento di un mercato hayeckiano privo di comportamento centrale e sorretto dal comportamento indipendente degli agenti, è operazione relativamente semplice" (Terna, 2000).

Un passo decisamente più complesso sarà quello di avere una perfetta corrispondenza con il reale, al fine di prevederne le dinamiche evolutive.

Appendice A

Aspetti regolamentari del MTA

Il contenuto di questa appendice analizza, nelle linee fondamentali, alcuni mecanismi di funzionamento che caratterizzano il mercato telematico azionario (MTA) italiano[1]. L'MTA è il comparto di mercato gestito da Borsa Italiana S.p.A. in cui si negoziano azioni, obbligazioni convertibili, *warrant*, diritti di opzione e azioni di OICR quotati in borsa.

La formazione del prezzo del prezzo teorico di asta

Nella fase di asta l'operatotre autorizzato può immettere nel sistema proposte a prezzo di asta, cioè senza limite di prezzo, oppure proposte a prezzo limitato. Il sistema può, pertanto, ricadere in una delle tre situazioni qui elencate:

- il prezzo teorico può essere determinato, viene validato e il titolo apre o i contratti vengono perfezionati;
- il prezzo teorico può essere determinato, ma non viene validato e il titolo ritorna in fase d'asta;
- il prezzo teorico non può essere determinato.

L'impossibilità a determinare il prezzo teorico si verifica quando:

a) entrambi i lati del book sono vuoti;

b) un lato del book è vuoto e in questo caso, se in asta di apertura, le proposte limitate presenti sull'altro passano direttamente alla fase continua mantenendo il prezzo indicato;

[1]Descritto nel regolamento emanato da Borsa Italiana S.p.A., disponibile gratuitamente al sito *www.borsaitalia.it* ed attualmente in vigore dal 1 luglio 2002.

c) un lato del book è vuoto e in questo caso, se in asta di apertura, le proposte al prezzo di apertura presenti sull'altro passano direttamente alla fase continua con prezzo uguale a quello di controllo del giorno precedente;

d) un lato del book è vuoto e, se in fase di asta di apertura, le proposte limitate presenti sull'altro passano direttamente alla fase continua mantenendo il prezzo indicato mentre quelle al prezzo di apertura assumono il miglior prezzo limite del proprio lato;

e) nel caso in cui il miglior prezzo denaro è inferiore al miglior prezzo lettera, le proposte, se in fase di apertura, passano in continua con il loro prezzo.

In apertura, qualora il prezzo non possa essere determinato le proposte vengono passate direttamente alla fase di negoziazione continua, mantenendo intatte le priorità temporali. Diverso il caso dell'asta di chiusura, in cui, qualora non sia stato determinato un prezzo teorico d'asta, o non sia stato valicato, le proposte di negoziazione, qualora sia stata specificata la modalità di esecuzione "valido sino a data specificata" sono trasferite automaticamente alla fase di pre-asta di apertura del giorno successivo, secondo le seguenti modalità:

a) con il prezzo e la priorità temporale della proposta originaria, se si tratta di proposte con limite di prezzo;

b) senza limite di prezzo e la priorità della proposta originaria, se si tratta di proposte senza limite di prezzo.

Determinazione del prezzo teorico di asta

Il prezzo teorico di asta viene ricavato grazie a quattro regole applicate in maniera consequenziale per ottenere l'individuazione univoca del prezzo di apertura. Il sistema provvede ad ogni immissione, modifica o cancellazione delle proposte, fino al termine della fase di pre-asta, a calcolare continuamente il prezzo teorico.

1. Il prezzo teorico è quello che consente la negoziazione della maggior quantità di titoli.

   Ciò implica:

   - calcolare, per ogni prezzo individuato per le proposte a prezzo limitato, la quantità complessiva in danaro e in lettera;
   - calcolare la quantità negoziabile a ogni prezzo (la minore tra la quantità in danaro e quella in lettera per quel livello del book);
   - rilevare la quantità negoziabile maggiore tra tutte.

Qualora due prezzi abbiano la maggior quantità negoziabile è necessario ricorrere alla seconda regola:

2. In caso di più prezzi che soddisfino il criterio del maggior numero di strumenti finanziari scambiati occorre identificare il prezzo teorico di asta come quello che produce il minor quantitativo non negoziabile relativamente alle proposte in acquisto o in vendita, aventi prezzi uguali o migliori rispetto a quello considerato.

   Ciò implica verificare, nel caso di più proposte con la stessa quantità negoziabile secondo la regola 1), quale livello di prezzo tra questi rende minima in valore assoluto la differenza tra quantità in lettera e quantità in danaro. Nel caso anche in questo caso non sia possibile arrivare ad un unico prezzo teorico si ricorre alla terza regola:

3. Il prezzo teorico di asta risulta corrispondente al prezzo più vicino al prezzo di riferimento dell'ultima seduta.

   In questo caso occorre valutare il prezzo che in valore assoluto differisce meno dall'ultimo prezzo di riferimento disponibile. In caso di ulteriore ambiguità si ricorre all'ultima regola:

4. Il prezzo teorico di asta corrisponde al maggiore tra quelli equidistanti.

La fase della validazione

In questa fase il prezzo teorico determinato con i criteri elencati precedentemente viene sottoposto ad una verifica per determinarne la validità.

- Il prezzo teorico di asta per essere ritenuto valido deve consentire la negoziazione di una percentuale prefissata (di norma pari al 20%) della quantità complessivamente presente per il segno prevalente.
- Il prezzo teorico è valido se la differenza tra il prezzi teorico di apertura o di chiusura e il prezzo di riferimento precedente non supera una percentuale prefissata, di norma pari ad uno scostamento del 10%.
- Il prezzo teorico di asta è ritenuto valido se la proporzione delle proposte a prezzo di asta sul totale delle proposte per quello strumento non supera una determinata percentuale prefissata.

La fase di apertura

Durante la fase dell'apertura il sistema effettua la conclusione delle automatica dei contratti abbinando le proposte di segno contrario presenti sul mercato, cancella le proposte con modalità ineseguibili e inoltra le proposte rimanenti alla successiva fase di negoziazione. L'attribuzione delle quantità in questa fase avviene in questo modo: alle proposte sul lato maggiore del

mercato, ordinate per prezzo e priorità temporale, viene assegnata l'intera quantità richiesta fino ad esaurimento della quantità totale disponibile sul lato minore del mercato. Le proposte a prezzo peggiore di quello di apertura vengono escluse dalla attribuzione delle quantità. Successivamente avviene l'abbinamento delle proposte: le proposte vengono ordinate per quantità attribuite decrescenti e abbinate secondo tale sequenza. Tutti contratti sono eseguiti al prezzo di apertura.

La fase di chiusura

Anche la fase di chiusura, in analogia con quella di apertura, è articolata in tre momenti:

- "pre-asta" in cui avviene la determinazione del prezzo teorico d'asta di chiusura;
- "validazione" in cui il prezzo teorico viene valicato;
- "chiusura" in cui avviene la effettiva conclusione dei contratti.

Durante la fase di negoziazione continua e la pre-asta di chiusura, gli operatori possono immettere proposte con o senza limite di prezzo caratterizzate dalla modalità "valide solo in asta di chiusura". Durante la fase di negoziazione continua le proposte non sono esposte sul mercato: sono conservate nel sistema per la loro partecipazione alla fase di presta e sono registrate secondo la priorità temporale determinata dall'orario di immissione. All'inizio della fase di pre-asta sono esposte con priorità temporale inferiore a quella delle proposte già presenti sul mercato e priorità temporale superiore a quella delle proposte immesse nella fase di pre-asta stessa.

Le proposte vengono automaticamente cancellate al termine della seduta per l'eventuale quantità ineseguita.

Sospensione temporanea di uno strumento finanziario

Qualora durante la negoziazione continua di uno strumento finanziario il prezzo del contratto in corso di conclusione superi uno dei limiti di variazione dei prezzi previsti da un apposito articolo (4.10.2) del Regolamento del mercato, la negoziazione continua dello strumento finanziario viene automaticamente sospesa per un intervallo la cui durata è stabilita nelle Istruzioni al regolamento.

Durante la sospensione temporanea della negoziazione non sono consentite l'immissione, la modifica o la cancellazione di proposte.

Allo scadere della sospensione temporanea le negoziazioni riprendono con le modalità della negoziazione continua, salvo diverse disposizioni della Borsa Italiana ai sensi dell'articolo citato.

Le limitazioni a cui si riferisce l'articolo citato e che fanno scattare la sospensione per i titoli azionari sono i seguenti:

- limite massimo di variazione dei contratti rispetto al prezzo di riferimento: + o - 10%.
- limite massimo di variazione dei prezzi tra due contratti consecutivi: + o -5%.

La durata dell'intervallo di sospensione automatica delle negoziazioni al superamento dei limiti di variazione dei prezzi è fissata in 5 minuti.

Appendice B

PHP e mySQL

B.0.1 Le basi di dati in MySQL

Nella costruzione di sistemi di gestione dati di tipo multi-utente, come nel caso dell'applicazione SUMWeb, si è renderà necessaria la realizzazione di un apposito data-base.

Attraverso questo strumento sarà possibile gestire i dati necessari per consentire ad "agenti umani" di prendere parte alla simulazione completamente artificiale di SUM. Per la realizzazone del data-base si è scelto come ambiente di riferimento quello rappresentato da MySQL, capace di creare e gestire basi di dati relazionali basati sul linguaggio SQL.

Gli oggetti creati con questo linguaggio rendono possibile la creazione e lo sviluppo di applicazioni basate sull'architettura *client-server*: la gestione dei dati avviene a livello centrale (lato *server*) consentendo la connessione ad applicazioni *client* per l'interrogazione, l'inserimento e la manipolazione delle informazioni.

Anche la simulazione SUMWeb rientra in questo schema, trattandosi di una classica applicazione *web* in cui il *client* è rappresentato da un *browser* che si connette al *web server* (nel caso in oggetto basato su APACHE) ed interagisce con questo basandosi sul protocollo *http - Hyper Text Transfer Protocol*.

E' il *web server*, a sua volta ad interagire con il *database server* per l'interscambio di dati. La presenza del *web server* fa sì che l'architettura finale adottata per la realizzazione dell'applicazione SUMWeb sia basata sui tre strati descritti.

Flessibilità, semplicità e conseguente velocità di esecuzione sono state le caratteristiche che hanno determinato l'adozione di un'applicazione basata su MySQL come data base in grado di soddisfare le esigenze dettate dall'architettura descritta.

MySQL, nato come clone di mSQL, implementa un linguaggio di SQL che si avvicina allo standard ANSI-SQL92 *"Entry level"* (nonostante svariate differenze ed estensioni ampiamente documentate). Grazie alle proprie doti

di potenza e flessibilità, unitamente all'essere disponibile con una licenza che ne rende gratuito l'uso anche per fini commerciali, MySQL ha avuto un successo via, via crescente proprio nelle applicazioni *web-based* basate sul sistema operativo Linux.

L'unica limitazione al linguaggio è costituita dal fatto che MySQL non gestisce le transazioni, ossia la possibilità di eseguire una serie di modifiche al database e lasciarle in sospeso finché non sono state eseguite tutte. Nonostante l'importanza di questo elemento, le molteplici caratteristiche positive di MySQL si sono dimostrate in grado di sopperire a questa lacuna.

MySQL non è un linguaggio di programmazione di tipo procedurale, ossia non esistono cicli e condizioni, ma solo comandi ed istruzioni per la manipolazione dei dati che consentono di creare tabelle, collegarle ed eseguire interrogazioni e modifiche su di esse. Le interrogazioni sono basate sull'uso di funzioni per l'esecuzione di calcoli e altre operazioni sui dati presenti nelle tabelle anche se talvota per questi compiti si ricorre a linguaggi di programmazione esterni.

Frequentemente MySQL viene utilizzato da programmi sviluppati in altri linguaggi e fornisce un'interfaccia standardizzata per l'interazione con *server*, specializzati per l'esecuzione di operazioni di manipolazione di dati (*database server*).

MySQL, come l'intero linguaggio SQL, si fonda su di una serie di semplici concetti che possono così riassumersi:

- si basa su una serie di dati semplici: stringhe, interi, "blob" (*Binary Large Object*, sequenze di dati non strutturate, simili a *file* che vengono memorizzati all'interno del *database*). *Array*, puntatori o altri tipi di dati strutturati non sono presenti nel linguaggio;

- i dati manipolati da MySQL vengono memorizzati in tabelle, suddivise a loro volta in *record*, detti anche righe. Ogni riga contiene una sequenza di elementi, detti campi, ciascuno dei quali caratterizzato da un nome ed un tipo definiti in sede di creazione della tabella;

- operando sul database è possibile selezionare tutte le occorrenze di un campo di una tabella, ovvero una colonna. Ogni colonna è composta di elementi dello stesso tipo;

- l'inserimento, la rimozione e la modifica dei dati avviene operando per riga mentre l'interrogazione dei dati avviene operando opportune selezioni per colonna.

Una caratteristica peculiare di una base dati MySQL è che le righe vengono inserite in maniera totalmente non ordinata. Ciò comporta che una riga deve essere selezionata sulla base di una chiave primaria, univoca per quel *record*. Un eventuale tentativo di inserire una nuova riga con lo stesso valore di

chiave primaria è destinato a fallire in virtù dell'attento controllo di unicità della chiave stessa eseguito da MySQL.

E' possibile suddividere i comandi disponibili in MySQL in tre gruppi principali che possono così catalogarsi:

- *Data Definition Language* (DDL): il linguaggio di definizione dati comprende tutti i comandi necessari per la definizione delle tabelle e degli altri elementi accessori nella gestione delle informazioni;
- *Data Manipulation Language* (DML): il linguaggio per la manipolazione dati comprende tutti i comandi necessari per l'interrogazione e la modifica dei dati contenuti nel data base;
- *Data Contol Language* (DCL): il linguaggio per il controllo dati comprende i comandi che consentono il controllo dell'accesso ai dati contenuti nel database.

B.0.2 Il linguaggio PHP

Nella ambito della programmazione *web* occorre distinguere la programmazione dal lato *server* da quella dal lato del *client*.

Quest'ultima risulta, destinata all'estensione delle funzionalità legate al *browser*, ha visto emergere come standard di fatto, universalmente adottato, il linguaggio di programmazione Java. L'adozione di questo linguaggio se, da un lato, ha reso possibile la creazione e l'uso di *applet*, ossia di piccole applicazioni che possono essere trasferite dalla rete ed eseguite dal browser, dall'altro, ha favorito lo sviluppo di una versione del linguaggio semplificato. Le applicazioni derivanti da quest'ultima versione, nota come Javascript, possono essere inserite direttamente in una pagina *web* in modo da essere eseguite immediatamente dal *browser client*.

Una simile divisione di ambiti di applicabilità caratterizza anche i due linguaggi di programmazione adottati a livello *server* dove l'adozione di Perl, il linguaggio generico utilizzato con i *web server*, ha visto successivamente la nascita e lo sviluppo di un linguaggio di *script* ben più semplificato ed ottimizzato (PHP), in grado di essere incorporato nelle pagine html ed essere eseguito sul *server*.

Il linguaggio PHP, acronimo di *Personal Home Page*, è nato come semplice interprete di un linguaggio di *scritping* incorporato in html. Raggiunta una certa notorietà con la diffusione delle varie versioni progressivamente rilasciate, PHP è giunto a riprendere aspetti di Perl, coniugandoli concetti tipici di Java e di Javascript.

Le sue caratteristiche sono il frutto di una attenta progettazione finalizzata alla definizione di un linguaggio di *scripting* espressamente progettato per la realizzazione di applicazioni web lato server, che possano essere realizzate ed eseguite in maniera semplice, immediata ed efficace.

Tra i punti di forza del linguaggio troviamo, innanzi tutto, la indiscussa semplicità che, unitamente alla ricchezza della libreria di funzioni già implementate, rende agevole scrivere applicazioni complesse con poche righe di codice.

Se questi elementi potrebbero non sembrare sufficienti a determinare il successo che il linguaggio sta riscuotendo, un ulteriore, forse decisivo, punto di forza deriva dalla profonda integrazione con le basi di dati. A questo riguardo, le librerie di PHP prevedono il supporto per una grande quantità di *server* per la gestione di *database* SQL. Non solo MySQL, che sarà utilizzato per l'applicazione oggetto di sperimentazione, ma anche basi di dati in mSQL, PostgresSQL, Oracle, Sybase, Informix e Intebase trovano in PHP l'interfaccia *web* ideale.

Anche il lato *web* risulta particolarmente sviluppato grazie al supporto dei principali protocolli utilizzati nell'*internet* (http, FTP, SMTP, IMAP, LDAP).

Se a questi aspetti si aggiunge la profonda integrazione con il *web server*

più diffuso a livello mondiale, APACHE, si riesce a comprendere i perché della scelta compiuta in questa elaborazione e, soprattutto, il perché di un così rapido successo per un linguaggio di *scripting* nato sotto dall'abbreviazione, *Personal Home Page*.

Appendice C

Codice

Le pagine che seguono contengono parte del codice Objective-C e Swarm dell'applicazione SUMWeb.

Per la parte Objective-C sono presenti i listati di:

- **AvatarAgent** (.h e .m);
- **BasicSumAgent** (.h e .m);
- **Book** (.h e .m);
- **Matrix2** (.h e .m);
- **ModelSwarm** (.h e .m).

Per la parte PHP/web, invece, sono presenti i listati di:

- **book.php**;
- **user.php**;
- **index.html**

Bibliografia

[1] APPLE COMPUTER, *Object-Oriented Programming and the Objective-C language*. Developer's library, 1999.

[2] BARUCCI E., *Teoria dei mercati finanziari*. il Mulino, Bologna, 2000.

[3] BARBER B.M., LOEFFLER D., *The darthboard column: second-hand information and price pressure*, Journal of financial and quantitative analysis, giugno 1993.

[4] BORSA ITALIANA S.P.A, *Regolamento dei mercati organizzati e gestiti dalla Borsa Italiana S.p.A*, http://www.borsaitaliana.it, 2002.

[5] BELTRATTI A., MARGARITA S., TERNA P., *Neural Network for Economic and Financial Modelling*. International Thomson Computer Press, 1996.

[6] CAPARRELLI F., *Una verifica empirica dell'ipotesi di efficienza debole del mercato di borsa*, Il Risparmio, n.2, 1986.

[7] CAPARRELLI F., *La borsa italiana e l'efficienza semiforte*, Il Risparmio, n2, 1989.

[8] CAPARRELLI F., *Economia dei mercati finanziari*. McGraw-Hill, Milano, 1998.

[9] CHEN S., LUX T., MARCHESI M., *Testing for non linear structure in an artificial financial market*, Journal of Economic Behavior and Organization, 2001.

[10] DE BONDT W.F., THALER R., *Does the stock market overreact?*, Journal of Finance, n.3, 1985.

[11] EINSTEIN A., INFELD L., *L'evoluzione della fisica*. Universale scientifica Boringhieri, Torino, 1965.

[12] FAMA E., *Efficient capital market: a review of theory and empirical work*, Journal of Finance, maggio 1970.

[13] GALLANT S., *Neural Network Learning and Expert Systems*. The MIT Press, 1993.

[14] GALLANT S., *Neural Network Learning and Expert Systems*. The MIT Press, 1993.

[15] GARBADE K., *Teoria dei mercati finanziari*. il Mulino, Bologna, 1989.

[16] GROSSMAN S., *The informational role of prices*, Mit press, Massachussetts, Boston, 1989.

[17] HAYEK F., *The use of Knowledge in society*. The American Economic Review, 35.

[18] HEBB D., *The organization of Behavior*. Wiley, 1949.

[19] HORGAN J., *The End of Science: Facing the Limits of Knowledge in the Twilight of the Scientific Age*. Broadway Books, 1997.

[20] HOLLAND J., *Adaptation in natural and artificial systems*. University of Michigan Press, 1975.

[21] LANGTON C., *Vita Artificiale*. Sistemi Intelligenti, n.2, 1992.

[22] LEBARON B., *Agent-Based computational finance: suggested readings and early research*, in Journal of Economic Dynamics and Control 24, pp 679-702, 2000.

[23] LUNA F., STEFANSSON B., *Economic Simulation in Swarm: Agent-Based Modelling and Object Oriented Programming*. Kluwer Academic Publishers Boston/Dordrecht/London, 2000.

[24] JOHNSON P., *ASM documentation*, http://artstkmkt.sourceforge.net, 2000.

[25] MARGARITA S., *Verso un "robot oeconomicus": algoritmi genetici ed economia*. Sistemi Intelligenti, n.3, pp. 421-459, 1992.

[26] MAKIEL B.G., *Efficient Market Hipothesis*, MacMillan, 1992.

[27] MINSKY M., PAPERT S., *Perceptrons: An Introduction to Computational Geometry*. MIT Press, 1969.

[28] KEYNES J. M., *Teoria generale dell'occupazione, dell'interesse e della moneta*. Utet, Torino, 1986.

[29] PARISI D., *Intervista sulle reti neurali. Cervello e macchine intelligenti*. il Mulino, Bologna, 1989.

[30] PARISI D., *Mente. I nuovi modelli della vita artificiale*. il Mulino, Bologna, 1999.

[31] PARISI D., *Simulazioni. La realtà rifatta nel computer*. il Mulino, Bologna, 2001.

[32] RABERTO M., CICCOTTI S., FOCARDI S.M., MARCHESI M., *Agent-based simulation of a financial market*, PHYSICA, 2001.

[33] ROSEMBLATT F., *Principles of Neurodynamics: Perceptrons and the Theory of Brain Mechanisms*. Spartan Press , 1961.

[34] ROSSI P., *La rivoluzione scientifica: da Copernico a Newton*. Loescher, Torino, 1973.

[35] RUMELHART D., MCCLELLAND J., *PDP. Microstruttura dei processi cognitivi*. il Mulino, 1991.

[36] RUMELHART D., MCCLELLAND J., *Parallel Distributed Processing: Explorations in the Microstructures of Cognition*. MIT Press, vol.2, 1986.

[37] SCIABARRA M., *Linux e programmazione WEB*, McGraw-Hill, Milano, 1999.

[38] SIAT, *Atti del convegno annuale*, Perugia, 2002.

[39] STEWART I., *Dio gioca a dadi?*. Bollati Boringhieri, Torino, 1993.

[40] TERNA P., *Reti neurali artificiali e modelli con agenti adattivi*. XXXV Riunione Scientifica annuale della Società italiana degli economisti, 1994.

[41] TERNA P., *Simulation Tools for Social Scientists: Building Agent-Based Models with Swarm*. Journal of Artificial Societies and Social Simulation, vol.1, n.2, 1998.

[42] TERNA P., *Creare mondi artificiali: una nota su Sugarscape e due commenti*. Sistemi Intelligenti, n.3, pp. 489-496, 1998.

[43] TESTFATSION L., *Introduction to the special issue on agent-based computational economics*, Journal of Economic Dynamics & Control, 25 (2001), pp.281-293, 2000.

[44] WALDROP M. M., *Complessità, Uomini ed Idee al confine tra Ordine e Caos*. Instar Libri, Torino, 1995.

[45] WOOLDRIDGE M.J., JENNINGS N.R., *Intelligents Agents: theory and practice*. Knowledge Engineering Review, vol.10, n.2, 1995.