ALMA MATER STUDIORUM
UNIVERSITÀ DEGLI STUDI DI BOLOGNA

II facoltà di Ingegneria - Cesena
Ingegneria Informatica

LA COMPUTAZIONE EVOLUTIVA NEI SISTEMI MULTI-AGENTE: SCENARI APPLICATIVI

Sistemi Multi-Agente L-S

Elia Maldini
matr. 0000346060

prof. Andrea Omicini

ANNO ACCADEMICO 2009/10

Indice

Introduzione

Nel corso del tempo la complessità dei sistemi computazionali è cresciuta sempre più. Ciò ha spinto la ricerca scientifica a proporre modelli e paradigmi più potenti che hanno permesso di dominare tali complessità. In molti casi, i concetti fondazionali di tali studi hanno preso ispirazione da comportamenti e abitudini osservabili in natura. In tale scenario l'uomo ha costituito il modello d'esempio principale, per la sue enormi capacità di risolvere problemi e le sue attitudini ad adattarsi all'ambiente.

Una ricostruzione artificiale del modo di pensare umano è rappresentato, ad esempio, dagli *Intentional System*. Il comportamento di tali sistemi è descritto, spiegato e predetto attribuendo loro caratteristiche tipiche della mente umana come ad esempio la conoscenza, i desideri e le capacità di ragionamento. Nei sistemi multi-agente l'adozione di questa astrazione si concretizza nella costruzione di agenti dotati di stati mentali e in grado di effettuare, su di essi, ragionamenti al fine di raggiungere obiettivi prefissati.

Nei sistemi multi-agente, tuttavia, le sole capacità computazionali dei singoli non sono sufficienti. Gli agenti si trovano immersi in un ambiente in cui condividono oggetti comuni e interagiscono fra loro. Tipicamente il comportamento assunto da ciascuna entità determina le performance dell'intero sistema. È necessario, quindi, che le parti in gioco collaborino fra loro per il raggiungimento degli obiettivi individuali e sociali. Anche questa problematica viene risolta individuando una possibile soluzione attraverso lo studio dei comportamenti umani.

L'*Activity Theory* è una teoria di tipo psicologico che studia le attività svolte dagli individui quando questi sono situati all'interno di contesti sociali. Secondo tale teoria ogni azione umana, sia questa comunicativa o svolta sull'ambiente circostante, è mediata da un *artefatto*; ovvero un dispositivo fisico o cognitivo in grado di estendere le capacità dell'individuo consentendogli di raggiungere i propri obiettivi. Nei sistemi multi-agente questa idea è stata adottata con l'introduzione di dispositivi denominati, appunto, artefatti attraverso i quali l'agente interagisce con l'ambiente e comunica con gli altri agenti. Ogni artefatto adempie ad una funzione ben precisa e rappresenta uno strumento che amplia le capacità dell'agente (in alcuni casi può anche ridurle). L'astrazione di artefatto gioca un ruolo importante anche nella coordinazione fra gli agenti infatti, in tale scenario, può essere intesa come una generalizzazione di medium di coordinazione in grado di abilitare e governare le interazioni fra gli agenti.

A differenza degli artefatti, gli agenti sono entità dotate di uno scopo. Essi

agiscono in funzione di un obiettivo che può essere rappresentato al loro interno in maniera esplicita o implicita. Nel primo caso si parla di agenti *goal-directed* ovvero entità intelligenti in grado di ragionare su una rappresentazione astratta di un obiettivo, al fine di individuare la sequenza di azioni per raggiungerlo. Nel secondo caso, invece, gli agenti *goal-oriented* si basano sulla rappresentazione di uno stato finale e attuano le politiche necessarie per raggiungerlo.

Una problematica che sorge a questo punto è la definizione del goal. Esistono problemi in cui è facile definirlo mentre in altri, come ad esempio nei problemi di ottimizzazione, risulta difficile formulare o ottenere in tempi ragionevoli la soluzione ottima del problema. L'approccio adottato, in tali casi, è quello di non risolvere all'ottimo il problema ma di cercare soluzioni di tipo euristico ovvero approssimazioni della soluzione ottima. In tale scenario una possibile strategia è suggerita dall'intelligenza artificiale attraverso le tecniche dette *metaeuristiche* che basano il funzionamento degli algoritmi risolutivi proposti, su fenomeni che si osservano in natura (es. metodi di approvvigionamento di cibo delle formiche, evoluzione delle specie).

La *computazione evolutiva* fa parte di queste tecniche e riproduce artificialmente i fenomeni di selezione, riproduzione e mutazione osservabili nell'evoluzione degli esseri viventi. L'agoritmo risolutivo proposto, parte da una popolazione di individui, che rappresenta l'insieme delle possibili soluzioni di un problema e sulla base di quest'ultima attua i meccanismi evolutivi che alterano le caratteristiche genetiche di ciascun individuo producendo una nuova popolazione. In questo modo l'algoritmo esplora lo spazio delle soluzioni e sulla base di un valore di fitness associato a ciascuna di esse, cerca la soluzione migliore del problema (non è detto che sia quella ottima). Questa tecnica viene impiegata anche nei sistemi multi-agente; (*Evolutionary Multi-Agent Systems*) in tale scenario due sono gli approcci possibili. Nel primo caso gli algoritmi evolutivi vengono impiegati da ciascun agente in maniera indipendente come metodo metaeuristico di risoluzione di problemi. Nel secondo caso, il più interessante, il problema è codificato in un obiettivo sociale da raggiungere. Ciascun agente costituisce la soluzione possibile del problema ed è soggetto ai meccanismi evolutivi dell'intero sistema i quali, ad ogni iterazione, determinano una nuova generazione di agenti che sostituisce quella vecchia. Tali meccanismi quindi vengono adottati come supporto al processo di ragionamento e apprendimento al fine di individuare la configurazione ottima del sistema.

Gli evolutionary multi-agent system non solo consentono di risolvere con il paradigma multi-agente problemi difficili ma ricorrendo ai meccanismi evolutivi rinforzano una delle più importanti caratteristiche degli agenti: l'autonomia. Infatti attraverso gli operatori genetici la ricerca del comportamento migliore da assumere o delle azioni da compiere può essere effetuata dagli agenti in modo autonomo, attraverso lo scambio reciproco del patrimonio genetico rappresentativo del comportamento migliore. In tale scenario il compito del progettista è quello di predisporre gli agenti ad evolversi e a riconoscere il comportamento migliore da adottare in funzione del contesto in cui sono immersi (concetto di fitness). Sarà poi il sistema stesso che autonomamente cercherà di individuare e raggiugere la soluzione migliore.

I vantaggi apportati dall'attribuzione ad un sistema di comportamenti tipi-

ci della natura umana si riscontrano nelle attività di sviluppo e manutenzione. In tale scenario, infatti, l'ingegnere o il programmatore progettano e implementano agenti in grado di agire in modo molto simile a come loro stessi farebbero se inseriti nel medesimo contesto.

Capitolo 1

Panoramica delle tecnologie

1.1 Computazione Evolutiva

La computazione evolutiva è un metodo di risoluzione di problemi che fa parte della famiglia degli approcci metaeuristici di tipo *population based*. Il funzionamento di tale approccio fa riferimento alla metafora naturale dell'evoluzione delle specie.

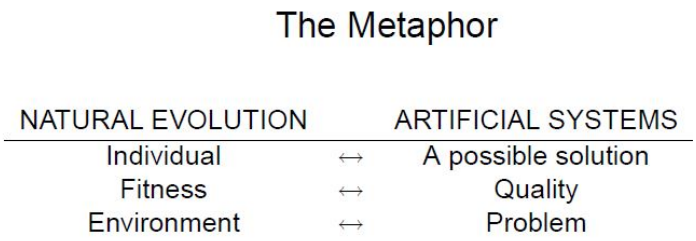

Figura 1.1: Confronto con la metafora naturale.

Come per tutti gli approcci metaeuristici population based, l'algoritmo parte da un'insieme di soluzioni candidate di partenza che costituiscono la popolazione iniziale e cerca di individuare la soluzione desiderata (o un'appriossimazione della stessa) alterando iterativamente le caratteristiche di tale popolazione.

In accordo con quanto accade in natura a ciascun individuo è associato un cromosoma che lo identifica. Tale cromosoma è composto da unità detti geni aventi un proprio dominio di esistenza.

All'interno del cromosoma è situata la codifica di una possibile soluzione del problema. In questo modo, alterando i valori assegnati ai geni si esplora lo spazio delle soluzioni.

Gli operatori messi a disposizione dalla computazione evolutiva sono:

- selezione
- ricombinazione

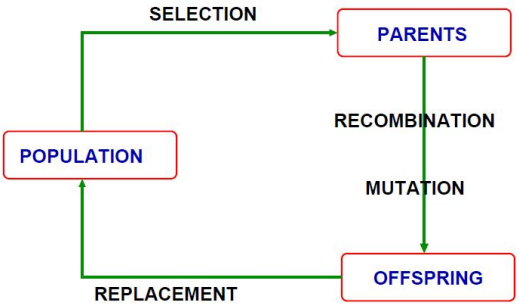

Figura 1.2: Ciclo evolutivo di una popolazione.

- mutazione
- sostituzione

Attraverso tali operazioni ad ogni iterazione dell'algoritmo si determina una nuova generazione di individui. La semantica degli operatori sopracitati dipende dal problema considerato.

1.1.1 Simple Genetic Algorithm

L'algoritmo genetico più semplice è quello che codifica ciscun individuo attraverso stringhe binarie. Ciascun gene rappresenta un bit che può assumere i valori binari 0 o 1. Il cromosoma è una stringa di bit che viene opportunamente alterata a seconda dell'operatore genetico considerato.

In tale scenario, la mutazione, altera (commuta) un gene scelto secondo una certa probabilità.

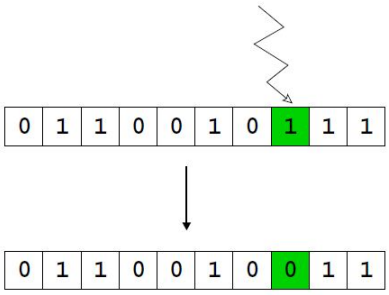

Figura 1.3: Operatore di mutazione.

La ricombinazione, invece, prende due individui e sulla base di un *crossover point* spezza la sequenza genetica dei due cromosomi e ne scambia l'ordine.

Ad ogni individuo è associato un valore di fitness che costituisce la qualità della soluzione del problema.

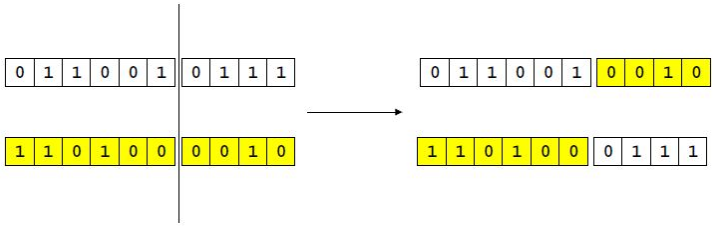

Figura 1.4: Operatore di crossover.

L'operatore di selezione preleva iterativamente gli individui della popolazione che adranno, poi, a ricombinarsi e a generare nuovi individui. La selezione avviene in modo proporzionale alla fitness, ovvero, tale operatore predilige individui caratterizzati da valori di fitness alti al fine di migliorare le soluzioni correnti e garantire la convergenza all'ottimo.

L'operatore di sostituzione, infine, si occupa di costituire di volta in volta la nuova generazione di soluzioni andando a sostituire i vecchi individui con quelli nuovi. L'algoritmo evolutivo termina nel momento in cui nella popolazione è presente la soluzione cercata. Tuttavia la terminazione è prevista anche qualora si sia raggiunto un limite di tempo o si è in una condizione detta *stagnation* per cui l'algoritmo tende a generare iterativamente la stessa popolazione.

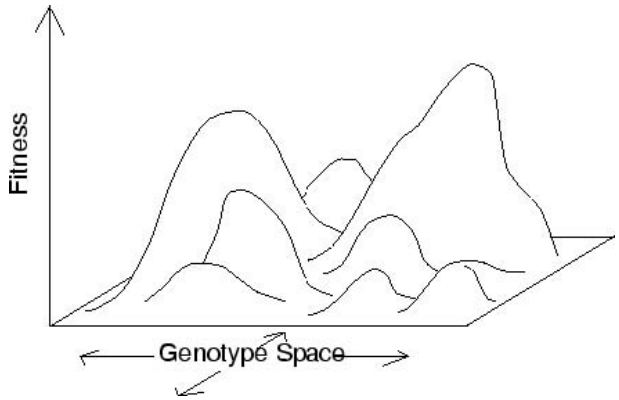

Figura 1.5: Campionamneto della fitness landscape.

Se si osserva il comportamento dell'algoritmo genetico in un grafico in cui viene riportata la distribuzione della fitness nello spazio delle soluzioni si nota che quello che esso effettua è un campionamento dell'intero spazio alla ricerca del picco più alto (individuo con più alta fitness oppure soluzione migliore del problema).Per capire quanto detto consideriamo un esempio: supponiamo di avere una popolazione quasi interamente formata da individui che si trovano su un massimo locale (si è giunti a questa situazione per particolari dinamiche di evoluzione della popolazione). Prima o poi,a causa di una mutazione o del processo di selezione, nella popolazione inizieranno ad essere presenti anche individui appartenenti al picco più alto. Dato che il processo di selezione predilige gli individui con maggior fitness, la nostra popolazione si sposterà pian pian dal massimo locale al massimo assoluto.

1.2 Sistemi Multi-Agente

Un sistema multi-agente è caratterizzato da un'insieme di entità interagenti e situate all'interno di un ambiente con la caratteristica principale di essere autonome. Una proprietà importante di un agente è quella di essere in grado di acquisire informazioni dall'ambiente e di eseguire azioni su di esso. Ovviamente le interazioni con l'ambiente e con le altre entità devono essere finalizzate al raggiungimento di un'obbiettivo. Esistono diversi modi per classificare un sistema multi-agente [1]:

- Sistemi distribuiti o supervisionati.
- Comunicazione esplicita o implicita.
- Eterogeneità

1.2.1 Sistemi Distribuiti e Supervisionati

Nella robotica, in particolare, un sistema si dice *supervisionato* se è presente un'entità centrale (es. computer, processore) in grado di raccogliere informazioni dai sensori e inviare comandi agli altri agenti per mezzo di attuatori. Lo svantaggio principale di tale architettura è che un mancato funzionamento del *supervisore* compromette le performance dell'intero sistema. Al fine di risolvere tale problematica, allora, si introducono i sistemi *distribuiti* caratterizzati da agenti indipendenti i cui comportamenti non sono dettati dalle informazioni ricevute da un'entità centrale ma agiscono in completa autonomia con un obbiettivo prefissato. I benefici apportati si traducono in una maggior robustezza ai guasti, poiché il sistema è in grado di supportare agilmente l'eventuale mancato funzionamento di un agente.

1.2.2 Comunicazione Esplicita o Implicita

Una seconda classificazione può essere effettuata considerando i possibili modi di comunicare degli agenti. A tal proposito si distinguono 3 diversi casi:

- Nessuna Comunicazione.
- Comunicazione Esplicita.
- Comunicazione Implicita.

Nel primo caso non è previsto alcuno scambio di informazioni fra gli agenti e quest'ultimi risultano completamente indipendenti.

Il secondo caso è quello più comune e prevede l'utilizzo di determinati protocolli per la comunicazione diretta fra gli agenti mediante scambio di dati.

L'ultimo caso, invece, non prevede una comunicazione diretta ma prende ispirazione dalla comunicazione indiretta tipica delle colonie di formiche. Secondo questo paradigma naturale le formiche comunicano fra loro rilasciando una sostanza sul terreno detta feromone. Questa tecnica può essere riprodotta nei sistemi multi-agente ricorrendo all'ambiente come mezzo di trasmissione di informazioni (comunicazione indiretta).

1.2.3 Eterogeneità

L'eterogeneità è utilizzata nei sistemi multi-agente per valutare le differenze fra gli agenti. Quando tutti gli agenti sono fisicamente identici ed in grado di svolgere le medesime attività il sistema si dice *omogeneo*. In caso contrario quando ciascun agente è dotato particolari capacità, il sistema si dice *eterogeneo*.

1.3 Sistemi Evolutivi Multi-Agente

1.3.1 Introduzione

I sistemi evolutivi multi-agente combinano il concetto di computazione evolutiva con l'idea di sistema modellato ad agenti [3]. In tale scenario l'approccio evolutivo può presentarsi sotto varie forme.

Nella maggior parte dei casi gli algoritmi evolutivi sono usati da ogni singolo agente in sostegno all'esecuzione di determinate attività (es. apprendimento, ragionamento) oppure per supportare la coordinazione fra gli agenti stessi.

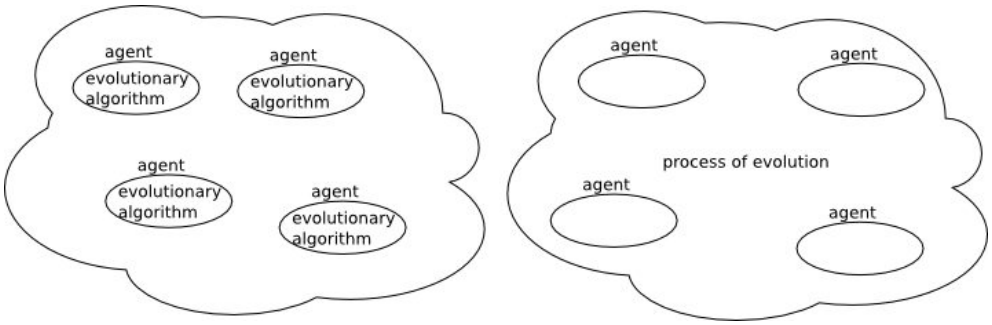

Figura 1.6: Approcci evolutivi nei sistemi multi-agente.

Tuttavia risultati interessanti possono essere osservati applicando il modello evolutivo fra gli agenti del sistema. In questo caso l'intero sistema è inteso come una popolazione ove ogni agente ne costituisce un individuo. Gli operatori genetici operano sull'intero sistema al fine di individuare la configurazione migliore o la soluzione del problema.

1.3.2 Aspetti Teorici

Gli operatori genetici di mutazione, ricombinazione e selezione applicati allo scenario multi-agente danno vita a due fenomeni principali:

- Morte
- Riproduzione

La prima azione ha come effetto l'eliminazione di un individuo dal sistema, mentre l'effetto della riproduzione è semplicemente la generazione di un nuovo agente da due genitori.

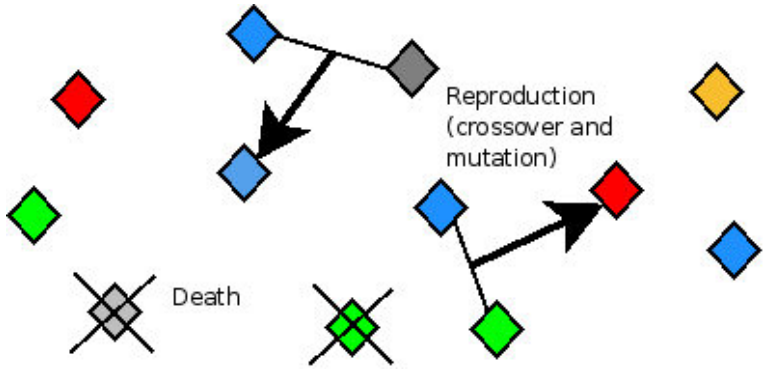

Figura 1.7: Riproduzione e la morte di agenti.

In accordo con quanto detto per la computazione evolutiva ogni individuo, o meglio agente è rappresentato dal proprio patrimonio genetico attraverso il cromosoma (genotipo). Durante il ciclo evolutivo questo patirmonio viene ereditato dalle nuove generazioni attraverso la mutazione e la ricombinazione. A fianco a questo patrimonio ciascun agente è dotato di una propria base di conoscenza che arricchisce durante tutto il ciclo di vita attraverso le percezioni che gli pervengono dall'esterno. Il comportamento (fenotipo) di ciascun agente è dettato proprio da questi due fattori, ovvero il patrimonio genetico e la conoscenza che ha dell'ambiente.

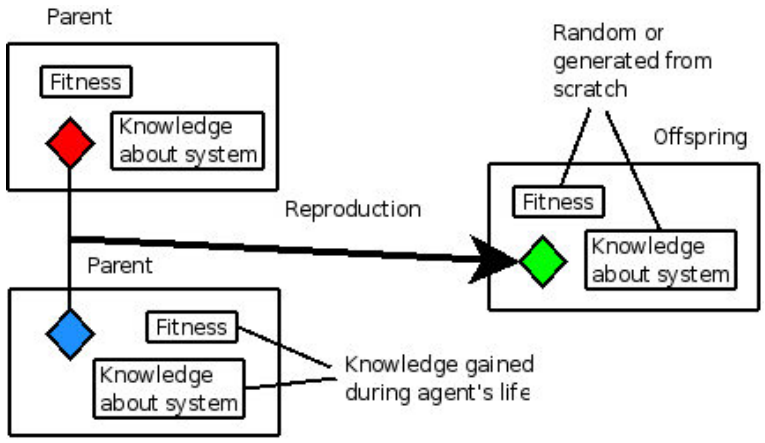

Figura 1.8: Valutazione della fitness.

Una caratteristica importante da osservare è che nei sistemi evolutivi multi-agente le attività svolte da ciascun agente sono circoscritte e non coinvolgono l'intero sistema. Per questo motivo, al fine di individuare le performance del sistema, non vi è la necessità di una valutazione globale del comportamento, ma è sufficiente che ciascun agente sia in grado di autovalutarsi in qualsiasi momento.

Da questo ne derivano le difficoltà nell'implementare l'aspetto del modello più importante: l'operatore di selezione. Tali difficoltà sono inoltre aggravate dal fatto che ogni agente è completamente autonomo e in grado quindi di

riprodursi in qualsiasi momento. Questo rende, quindi, difficile valutare il comportamento di tutti gli individui nello stesso momento.

Una delle possibili soluzioni a tale problema si basa sull'esistenza di una risorsa non rinnovabile denominata *energia vitale*. Tale energia viene guadagnata o persa dall'agente a seconda delle azioni eseguite sull'ambiente. In questo scenario, quindi, l'incremento dell'energia è la conseguenza per azioni *buone*, mentre la perdita costituisce la penalità pagata per un comportamento *cattivo* (la scelta su quali siano le azioni buone o cattive dipende dal problema da risolvere).

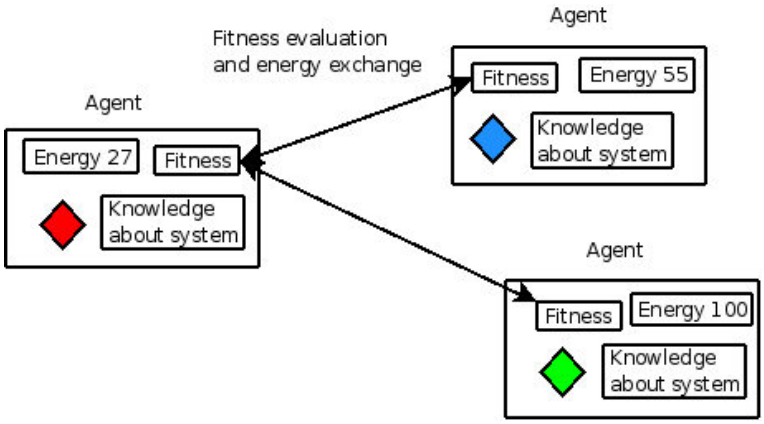

Figura 1.9: Sambio di energia fra agenti.

Allo stesso tempo il livello di energia vitale determina le azioni che l'agente è in grado di eseguire. Ad esempio un basso valore di energia può incrementare le possibilità di morte mentre una alto valore aumenta le probabilitá di riproduzione.

Esistono, infine, determinati algoritmi che stabiliscono come l'energia deve modificarsi durante la vita dell'agente. La variazione di tale energia e quindi gli algoritmi stessi sono spesso legati alla valutazione della fitness e allo scambio di energia con gli agenti vicini.

1.3.3 Modello Distribuito

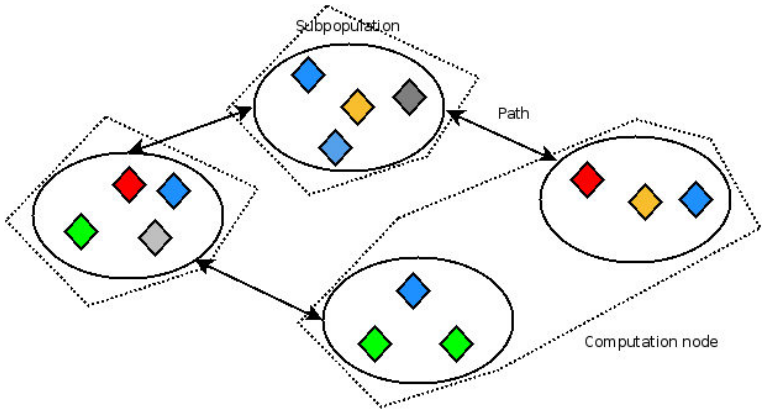

Figura 1.10: Modello distribuito di EMAS.

Le caratteristiche del paradigma multi-agente permettono, anche, di concepire sistemi come distribuiti in una rete. In tale scenario gli agenti e l'ambiente stesso sono allocati su più nodi di rete con una topologia ben definita.

Anche gli evolutionary multi-agent system sono compatibili con lo scenario distribuito. La popolazione di individui può essere facilmente decomposta in sotto-popolazioni ciascuna delle quali situata in un nodo della rete. Ogni sottopopolazione può attuare i processi evolutivi in modo indipendente e inoltre, implementando protocolli di comunicazione ad-hoc, può scambiare materiale genetico con altre sottopopolazioni.

Capitolo 2

Fully Distributed Evolutionary Robot

Figura 2.1: Esperimento

2.1 Ipotesi

Nell'esperimento denominato *Fully Distributed Evolutionary Robot* [1] ciascun robot è considerato un agente e la popolazione è omogenea, ovvero tutti gli individui sono dotati delle stesse capacità: acquisire informazioni, effettuare azioni, comunicare ed elaborare. Il sistema è completamente distribuito poiché non esiste un'entità centrale e tutte le informazioni necessarie sono distribuite fra i robot. Il tipo di comunicazione adottato è quello esplicito mediante scambio di dati.

Il compito di ciascun robot è quello di individuare autonomamente una strategia efficiente per evitare gli ostacoli. A tal proposito vengono utilizzati gli algoritmi evolutivi come metodi per l'autoapprendimento Ciascun robot è dotato di 16 sensori ad infrarosso con la funzionalità di emettitore e 8, invece, come ricevitori. I medesimi sensori vengono impiegati per la comunicazione fra gli agenti e per questo è previsto un protocollo opportuno che differenzia la percezione di un altro robot con la percezione di un'ostacolo.

2.2 Funzionamento

I possibili input del robot forniti dai sensori di prossimità sono codificati in 5 stati riassunti nella seguente tabella:

Stato1	Nessun Ostacolo
Stato2	Ostacolo a sinistra
Stato3	Ostacolo a destra
Stato4	Ostacolo frontale
Stato5	Intrappolato

A fronte dei possibili input il comportamento del robot (azioni possibili) è riassunto nella seguente tabella:

Azione1	Vai Avanti
Azione2	Gira a destra
Azione3	Gira a sinistra
Azione4	Torna Indietro

In accordo con il modello proposto dalla computazione evolutiva, ciascun individuo è caratterizzato da un cromosoma che codifica il comportamento assunto, attraverso l'associazione di un'azione per ogni possibile input. Il cromosoma è formato da N parole che corrispondono agli N stati in cui il robot può trovarsi. L'azione adottata, per ciascuno di essi, è encodificata negli M bit di cui ogni parola si compone. Il comportamento di ciascun robot descritto da tale cromosoma si evolve, quindi, secondo gli operatori genetici previsti.

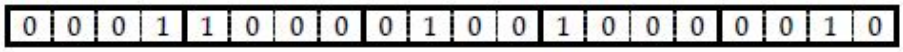

Figura 2.2: Esempio di cromosoma.

2.3 Operatori Genetici

2.3.1 Inizializzazione

Nella fase iniziale i robot assumono un comportamento non predicibile poichè i cromosomi vengono valorizzati in modo casuale.

2.3.2 Valutazione della Fitness

Nel caso più generale la valutazione del comportamento di ogni singolo individuo avviene ad opera di un supervisore il quale assegna i valori di fitness in base alle performance di ciascun robot. Nel caso considerato, per l'ipotesi di un sistema completamente distribuito, non sono previste entità centrali e ogni robot, quindi, è in grado di auto-valutare le proprie performance.

La valutazione della fitness di ogni individuo avviene attraverso la seguente equazione:

$$R_{N(1)}(i) = (1 - \alpha(i))R_{N(1)-1}(i) + \alpha(i)F_{N(i)}(i) \tag{2.1}$$

- $N(i)$ è il numero di passi temporali trascorsi dall'inizio della stima.
- $R_{N(i)}(i)$ è il guadagno stimato al tempo N(i).
- $F_{N(i)}(i)$ è il guadagno istantaneo al tempo N(i).

(Per *guadagno* si intende la distanza percorsa da un robot in un intervallo di tempo)

2.3.3 Crossover

Le condizioni sufficienti perchè due robot possano effettuare l'operazione di crossover sono le seguenti:

- I robot devono essere ad una distanza tale da permettere loro di comunicare
- Entrambi gli individui non devono aver effettuato altre operazioni genetiche nel breve tempo.

Tale operazione, come abbiamo visto, non garantisce la convergenza del sistema alla soluzione ottima specialmente se la popolazione di individui ha cardinalità bassa. Il rischio è quello di veder convergere il sistema ad una soluzione sub ottima ovvero un massimo locale.

A tal proposito si rende necessario l'operatore di mutazione che permette di generare, in modo casuale, sequenze di geni. L'effetto che si desidera da tale operazione è quella di allontanare la ricerca dai massimi locali fino a convergere alla soluzione ottima cercata.

2.3.4 Mutazione

L'approccio classico per realizzare la mutazione è quello di cambiare in maniera casuale il patrimonio genetico di un individuo. Lo svantaggio di questa politica è che le modifiche effettuate possano portare ad un cromosoma caratterizzato da valore di fitness più basso. (in parole povere la mutazione peggiora la soluzione).

Per evitare tali situazioni le condizioni di applicazione di tale operatore sono le seguenti:

- Il robot non deve aver effettuato di recente un'altra mutazione
- Il valore di fitness dell'individuo deve essere basso

La prima condizione assicura che il robot abbia tempo sufficiente per stimare correttamente la propria fitness, la seconda condizione, invece, impedisce di mutare una buona soluzione in una peggiore.

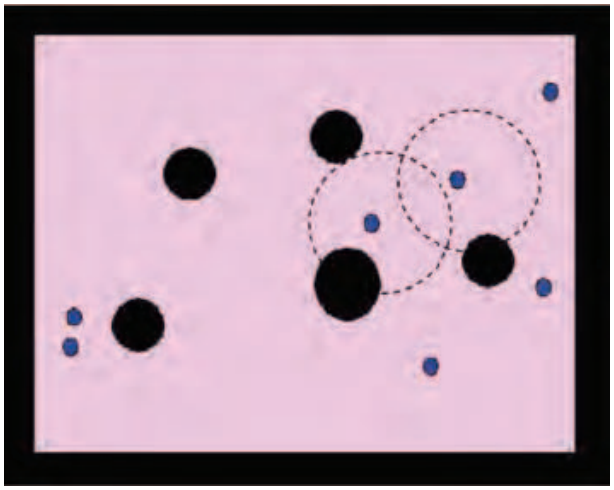

Figura 2.3: Situazione iniziale esperimento.

2.4 Simulazione

La simulazione prevede l'utilizzo di 7 robot (cerchi piccoli) in un ambiente costituito da 5 ostacoli (cerchi grandi). I robot devono auto-apprendere la strategia migliore per vivere in tale ambiente, devono cioè essere in grado di individuare le mosse migliori da effettuare, in funzione delle possibili condizioni in cui si possono trovare.

I possibili comportamenti di un robot, osservati durante la simulazione, sono i seguenti:

- Esecuzione dell'azione stabilita dal proprio patrimonio genetico.
- Aggiornamento del proprio stato a seconda delle percezioni.
- Mutazione.
- Crossover.

Un'importante questione di cui tenere conto è l'impostazione dei parametri di simulazione.

Quello di più cruciale importanza è la frequenza con cui un' individuo può effettuare le mutazioni: se tale valore è basso l'intero spazio delle soluzioni è esplorato più velocemente ma la stima della fitness è soggetta ad errori. Al contrario se la frequenza è bassa si ha un'esplorazione più lenta delle possibili soluzioni e la qualità di quest'ultime è valutata in modo migliore.

Un altro parametro importante è la dimensione della popolazione, ovvero il numero dei robot. Intuitivamente più agenti ci sono e più velocemente viene individuata la soluzione ottima. Ad esempio per il caso considerato, questo sarebbe vero se la condizione di convergenza è quella di avere almeno un agente, al termine della simulazione, con il comportamento desiderato. Nell'esempio proposto invece, la condizione di terminazione è che tutti gli individui raggiungano il comportamento previsto.

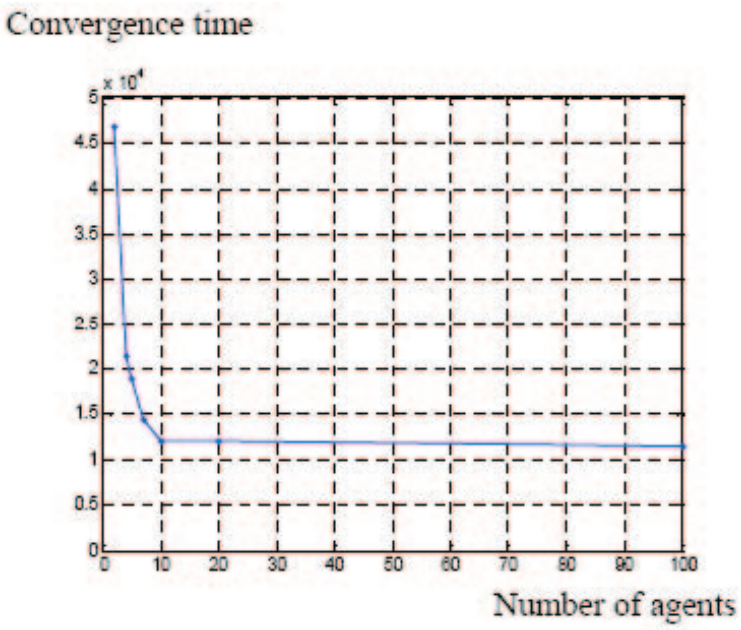

Figura 2.4: Situazione finale esperimento.

Nel grafico è rappresentata la relazione fra numero di agenti e tempo di convergenza e come è facile notare il tempo di convergenza decresce all'aumentare del numero di agenti fino alla soglia 10 dopo il quale il tempo, invece, diviene quasi costante indipendentemente dalla cardinalità della popolazione. Il motivo di questo comportamento risiede nel fatto che all'aumentare del numero di robot la soluzione ottima viene, sì individuata velocemente ma è richiesto comunque del tempo per propagare tale soluzione all'intera popolazione. È necessario quindi settare i parametri iniziali a seconda delle caratteristiche della simulazione e delle performance che si vogliono ottenere.

L'evoluzione del sistema, durante la simulazione, può essere riassunto nei seguenti passi:

1. Nella fase iniziale i robot adottano una strategia casuale dettata dallo stato iniziale del loro cromosoma. In tale situazione possono occorrere collisioni e i robot possono finire intrappolati. Il primo operatore ad essere eseguito è la mutazione a causa della bassa fitness che caratterizza ciascun individuo. Tale processo termina quando nei cromosomi degli individui è presente il comportamento : stato= *Intrappolato* azione= *Torna Indietro*.

2. A questo punto gli agenti sono liberi di muoversi nell'ambiente ed è molto probabile che almeno uno di loro abbia nel proprio cromosoma l'azione *Vai Avanti* associata per lo stato *Nessuno Ostacolo*. Questo individuo è quello che percorre più strada all'interno dell'ambiente e propaga in questo modo il proprio cromosoma agli altri agenti.

3. Nel momento in cui più di un agente è in grado di muoversi agevolmente nell'ambiente più operazioni di crossover hanno luogo. In questo modo la politica migliore si distribuisce a più di un robot.

4. Nel momento in cui un robot individua il comportamento ottimale è in grado di muoversi in tutto l'ambiente e incontrare così tutti gli altri agenti. In questo modo trasmette il suo cromosoma agli altri e ciascun individuo della popolazione inizia ad assumere il comportamento migliore.

2.5 Risultati

La situazione del sistema alla fine dell'esperimento è rappresentato nella figura sottostante.

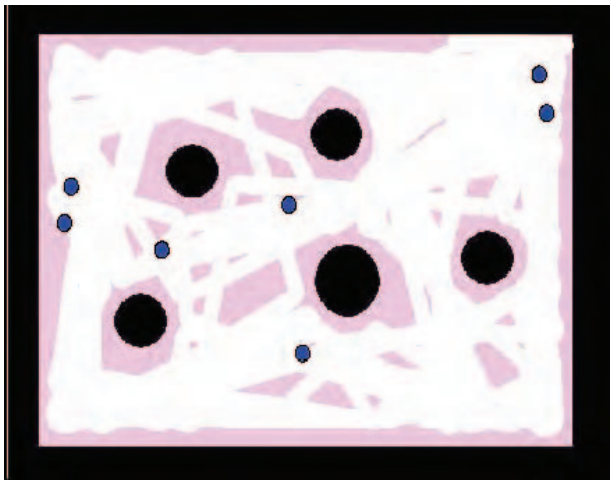

Figura 2.5: Situazione finale esperimento.

La soluzione ottima, rappresentata nella tabella, è ottenuta indipendentemente dalle condizioni iniziali e dalla struttura dell'ambiente; la convergenza è garantita in tempi ragionevoli.

1:Nessun ostacolo	1:Vai Avanti
2:Ostacolo a sinistra	2:Gira a destra
3:Ostacolo a destra	3:Gira a sinistra
4:Ostacolo frontale	2 o 3:Gira a destra o a sinistra
5:Intrappolato	4:Torna indietro

Dall'esperimento sono emersi due fenomeni molto importanti:

- Il robot che giunge per prima alla strategia più efficiente ha elevata probabilità di appatenere alle generazioni future, egli si muove in modo lento ed uniforme nell'ambiente incrementando, di fatto, la probabilità di incontrare gli altri robot. In questo modo diviene il candidato più popolare per la riproduzione (crossover).

- Il sistema è robusto ai *fault*. Gli individui che per qualche motivo cessano di funzionare non influenzano il comportamento degli altri individui. A sottolineare questa caratteristica, durante l'esperimento sono stati aggiunti e rimossi dei robot dalla popolazione senza osservare effetti negativi sul comportamento complessivo.

Infine, un fattore di cui tenere considerazione è la dimensione dell'ambiente che dipende fortemente dal numero di individui presenti. Esiste una relazione di proporzionalità fra le due grandezze che pone un limite massimo al numero di robot che possono essere posti all'interno di un dato ambiente.È stato verificato sperimentalmente che all'aumentare degli individui sopra ad una certa soglia il tempo di convergenza non decresce più.

Capitolo 3

Full Dispersion Game

3.1 Scenario

Mediante il *Full Dispersion Game* [2] si considera il problema in cui N agenti devono concorrere per l'assegnazione di N task da eseguire. Non sono previste entità centralizzate quindi il sistema risulta completamente distribuito.

Ciascun agente $i(1 \leq i \leq n)$ è rappresentato da una popolazione di individui pop_i; ogni individuo è encodificato in un cromosoma che esprime la preferenza dell'agente nell'assegnamento dei task. La struttura del cromosoma consiste in un insieme di geni $g_j(1 \leq j \leq n)$ pari al numero dei task fra i quali l'agente deve scegliere. Per un dato agente la probabilità di scegliere il task j al passo k è dato dalla seguente formula (Boltzmann):

$$\frac{e^{\frac{g_j(k)}{\tau}}}{\sum_{l=1}^{n} e^{\frac{g_l(k)}{\tau}}} \tag{3.1}$$

Nell'insieme dei possibili comportamenti assumibili, ovviamente solo uno deve essere quello dominante. A tal proposito il cromosoma avente fitness più alta stabilisce il comportamento vero e proprio adottato.

3.2 Operatori Genetici

3.2.1 Inizializzazione

Nella fase iniziale i cromosomi appartenenti agli individui di tutte le popolazioni vengono inizializzati a zero. In questo modo, perciò, ogni individuo ha un'equiprobabile preferenza per tutti i task disponibili. Il sistema, così, inizializzato presenta un comportamento del tutto casuale. Per garantire un assestamento del sistema in tempi brevi il valore di fitness associato a ciascun cromosoma viene settato ad un valore prefissato (tipicamente 0.5).

3.2.2 Valutazione della Fitness

L'idea generale che sta alla base del calcolo della fitness è la seguente:

> *La fitness associata ad un cromosoma c_i della popolazione i corrisponde alla probabilità che l'azione specificata dal cromosoma stesso venga eseguita. Tale probabilità dipende dalle scelte effettuate dagli altri agenti.*

La funzione che approssima tale idea, calcola la fitness del cromosoma c_i della popolazione pop_i in diversi passi:

1. Vengono individuati i cromosomi c_l con $l \neq i \wedge 1 \leq l \leq n$ caratterizzati dalla fitness più alta per ciascuna popolazione pop_l. (In questo modo si è a conoscenza della preferenza di ciascun agente nella scelta del task da eseguire)

2. Il task scelto da un agente j (rappresentato dal cromosoma c_j) è assegnato in modo casuale ad uno dei contendenti di tale task. Si procede in questo modo per ogni agente j con $j \neq i$.

3. Al termine di tali operazioni, il valore della fitness del cromosoma c_i cercato vale 1 se il task (del quale esprime preferenza) è assegnato al corrispondente agente, 0 in caso contrario.

Una più accurata approssimazione del calcolo della fitness è ottenuto eseguendo più cicli di tale algoritmo. In questo modo si hanno a disposizione più risultati distinti (ciascun ciclo prende il nome di *sample*) dei quali è possibile, poi, eseguire la media.

Nota la fitness associata a ciascun cromosoma e quindi nota la preferenza che ciascun agente ha nella scelta del task da eseguire, è possibile determinate la fitness dell'intero sistema. Quest'ultima può assumere un valore compreso tra 0 e 1 ed è proporzionale al numero di task che sono eseguiti da almeno un agente. Il valore di fitness pari a 1 è raggiunto se e solo se tutti gli agenti scelgono un task distinto.

Il calcolo della fitness viene tipicamente utilizzato nella fase iniziale dell'operazione di replacement in cui si deve aver certezza che gli individui figli siano migliori degli individui padre.

3.2.3 Crossover

L'operazione di crossover non è prevista, ogni popolazione è da intendersi come una specie distinta poiché non viene scambiato patrimonio genetico da generazione a generazione. Come si vedrà successivamente, la scelta degli individui padre della nuova generazione è realizzata in modo casuale.

3.2.4 Mutazione

La produzione della nuova generazione di individui avviene esclusivamente attraverso la mutazione. Ogni individuo figlio viene, quindi, creato come copia

del padre ed è soggetto a variazioni casuali del patrimonio genetico. Tale operazione avviene scegliendo casualmente un gene dal cromosoma padre e sommando a tale grandezza una quantità $e \in [-\alpha, \alpha]$. Il parametro α viene usualmente settato a 0.5.

3.2.5 Replacement

L'esecuzione dell'algoritmo genetico termina con la fase di replacement nel quale i nuovi individui generati vanno a costituire la nuova generazione prendendo il posto, dei vecchi individui. Come detto precedentemente siccome non è previsto l'operazione di crossover, gli individui padre della nuova generazione vengono scelti casualmente. Al fine di garantire la convergenza alla soluzione ottima, la sostituzione della vecchia generazione con quella nuova avviene solo se la fitness degli individui padre è inferiore a quella dei figli. Questo garantisce il miglioramento della soluzione ad ogni iterazione dell'algoritmo.

3.3 Osservazioni

Durante il processo evolutivo del sistema si possono osservare due tipologie differenti di computazione:

Parallela: Il processo di generazione dei nuovi individui e il successivo aggiornamento della popolazione viene eseguito in contemporanea da tutti gli agenti. La generazione dei nuovi individui calcola la propria fitness sulla base delle caratteristiche degli individui dell'epoca precedente. In questo modo le popolazioni evolvono parallelamente.

Sequenziale: Ad ogni passo dell'algoritmo in ogni popolazione (agente) la vecchia generazione lascia il posto a quella nuova, ovvero terminato l'aggiornamento della popolazione pop_i la nuova generazione pop_{i+1} scaturisce dall'evoluzione della generazione iniziale. In tal senso la computazione è sequenziale.

3.4 Risultati

La risoluzione del problema del full dispersion game mediante un sistema evolutivo multi-agente è stata eseguita ed analizzata sotto diverse condizioni iniziali. I parametri di simulazione presi in considerazione sono la dimensione della popolazione S di ciascun agente e il numero di individui figli C prodotti in ogni generazione. Nella simulazione effettuata si suppone, sempre, S=C e vengono di seguito proposti i risultati per S=C pari a 1, 5 e 10.

Nei grafici viene mostrato l'andamento della fitness dell'intero sistema nel caso in cui questa venga calcolata attraverso la media di *samples*. A tal fine vengono mostrati le differenze dei risultati per 1, 10 e 100 samples.

Come detto in precedenza, la soluzione ottima del problema corrisponde all'assegnamento di un agente per ciascun task e a tale soluzione corrisponde una fitness del sistema pari a 1.

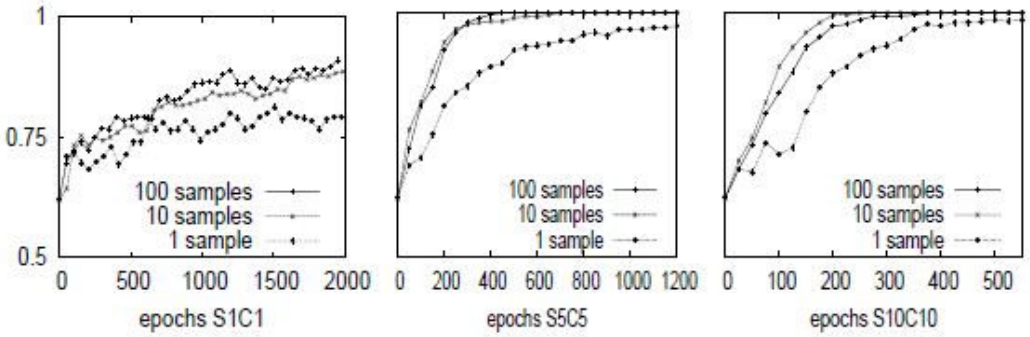

Figura 3.1: Risultati simulazione effettuata con 10 agenti.

Come si può vedere dai grafici, per S=C=1 il sistema mostra difficoltà a convergere alla soluzione ottima e anche nel caso in cui la fitness (di ciascun cromosoma) venga calcolata con un'approssimazione migliore (100 samples) i risultati non sono, comunque, soddisfacenti.

Per S=C=5 il sistema assume il comportamento migliore soprattutto nel caso in cui la fitness venga calcolata attraverso 10 e 100 samples. In tale situazione al sistema occorrono pochi cicli evolutivi per convergere alla soluzione ottima.

Infine, come si può vedere dall'ultimo grafico, i benefici apportati da un eventuale ampliamento della popolazione e degli individui figli a 10 (S=C=10) non porta sostanziali miglioramenti.

Capitolo 4

Conclusioni

La computazione evolutiva si può manifestare nei sistemi multi-agente in due forme.

Nel primo caso gli algoritmi evolutivi vengono utilizzati all'interno di ciascun agente come metodo metaeuristico di risoluzione di problemi. Fa parte di questa categoria l'esempio proposto del *Full Dispersion Game*. In tale esperimento ciascun agente è sede di una popolazione distinta che rappresenta, attraverso gli individui di cui si compone, la propensione nella scelta di un task. L'algoritmo parte da uno stato iniziale in cui gli agenti esprimono un'equiprobabile preferenza per tutti i task e ad agni iterazione, attraverso l'opportuna applicazione degli operatori genetici tale scelta viene sempre più raffinata. In questo caso quindi la computazione evolutiva è utilizzata da ciascun agente per risolvere il problema dell'assegnameto dei task e come si è potuto verificare sperimentalmente attraverso opportune condizioni iniziali, conduce alla soluzione ottima del problema.

Nel secondo caso, invece, gli agenti sono gli individui di una popolazione e sono soggetti agli algoritmi evolutivi che, a differenza di quanto accade per lo scenario precedente, vengono applicati all'intero sistema al fine di alterare, iterativamente, le caratteristiche genetiche di ciascun agente. L'esempio del *Fully Distributed Evolutionary Robot* mette in evidenza questi meccanismi. Ciascun robot racchiude un cromosoma che codifica il comportamento da assumere in funzione delle possibili condizioni in cui si può trovare. Gli algoritmi evolutivi permettono ai robot di modificare tale cromosoma e scambiarsi il patrimonio genetico al fine di supportare il processo di apprendimento e decisione. Anche in questo caso i risultati sperimentali evidenziano l'efficacia dell'algoritmo nell'ottenere la soluzione ottima del problema.

Come dimostrano i risultati ottenuti, quindi, gli algoritmi evolutivi applicati ai sistemi multi-agente sono in grado di portare benefici che possono essere misurati in termini di miglioramento dei processi decisionali e di apprendimento. Inoltre in caso di problemi difficili da trattare costituisco un possibile approccio euristico per ottenere una soluzione.

Gli evolutionary multi-agent system rinforzano anche la caratteristica principale degli agenti, ovvero l'autonomia. Nel secondo esempio trattato, infatti, ogni robot determina autonomamente il comportamento migliore da assumere per ogni situazione in cui si trova. Il progettista quindi ha, esclusivamente, il

compito di rendere ciascun robot in grado di compiere i meccanismi genetici lasciando poi alla dinamica del sistema di far apprendere a ciascuno di essi il comportamento migliore.

Bibliografia

[1] P. Lucidarme: *Evolutionary computation of multi-robot/agent systems* ARS Robotic Books, Chapter 16, I-Tech Education and Publishing, 2008.

[2] P. J. t Hoen and E. D. De Jong: *Evolutionary multi-agent systems*, 8th International Conference on Parallel Problem Solving from Nature, 2004

[3] Krzysztof CETNAROWICZ, *http://age.iisg.agh.edu.pl/emas/*, Evolutionary Multi-Agent System

[4] Hong Liu, Mingxi Tang, *Evolutionary design in a multi-agent design environment*, Design Technology Research Center, School of Design, The Hong Kong Polytechnic University 2005

Elenco delle figure