Capitolo 1

Algoritmi Genetici

I GA sono metodi adattativi che possono essere usati per risolvere problemi di ricerca e ottimizzazione. Sono basati sui processi genetici degli organismi biologici. Imitando questi processi, gli algoritmi genetici sono in grado di evolvere soluzioni per problemi del mondo reale, se sono stati codificati opportunamente. I principi di base dei GA sono stati definiti per la prima volta da Holland: essi simulano quei processi che nelle popolazioni naturali sono essenziali per l'evoluzione.

In Natura, gli individui di una popolazioni competono uno con l'altro per risorse come cibo, acqua e territorio. Inoltre membri della stessa specie spesso competono per attrarre un compagno. Quegli individui che hanno più successo nella sopravvivenza e nella riproduzione avranno un numero relativamente grande di discendenti.

I GA usano una diretta analogia con il comportamento della natura. Lavorano con una popolazione di individui, ciascuno dei quali rappresenta una possibile soluzione del problema posto. A ogni individuo è associato un punteggio di adattamento "*fitness score*" a seconda di quanto sia buona la soluzione al problema. Gli individui migliori hanno la possibilità di riprodursi incrociandosi con altri individui della popolazione. Questo produce nuovi individui discendenti che condividono alcune caratteristiche di ciascun genitore. Gli individui meno adattati hanno meno probabilità di riprodursi e quindi si estinguono. Una intera nuova popolazione di possibili soluzioni è così prodotta dalla selezione degli individui migliori della generazione corrente che accoppiandosi tra loro producono un nuovo insieme di individui. Questa nuova generazione contiene una proporzione più alta delle caratteristiche possedute dagli individui buoni della precedente generazione. In questo modo dopo molte generazioni le buone caratteristiche vengono propagate a tutta la popolazione, essendo mischiate e scambiate con altre buone caratteristiche. Favorendo l'accoppiamento tra gli individui più adatti, vengono esplorate la aree più promettenti dello spazio di ricerca.

I GA non garantiscono di trovare una soluzione ottima per un problema, ma generalmente trovano una soluzione sufficientemente buona e in tempi sufficientemente rapidi. Dove esistono tecniche specializzate per risolvere particolari problemi queste hanno spesso prestazioni migliori dei GA sia in termini di accuratezza che di velocità . Il terreno migliore dei GA sono dunque le aree dove non esistono tecniche specializzate. Dove esistono tecniche che funzionano bene, si possono avere miglioramenti "*ibridizzandole*" con i GA.

1.1 Caratteristiche dei GA

Gli Algoritmi Genetici sono metodi adattativi che possono essere usati per risolvere problemi di ricerca e ottimizzazione, basati sui principi di Darwin [Darwin, 1859]. Lavorano con una popolazione di possibili soluzioni di un dato problema. Ogni individuo all'interno della popolazione rappresenta una particolare soluzione del problema, generalmente espressa in una qualche forma di codice genetico. La popolazione evolve, per generazioni, in modo da produrre soluzioni migliori. Una schematizzazione dell'algoritmo è mostrata in Figura 1.

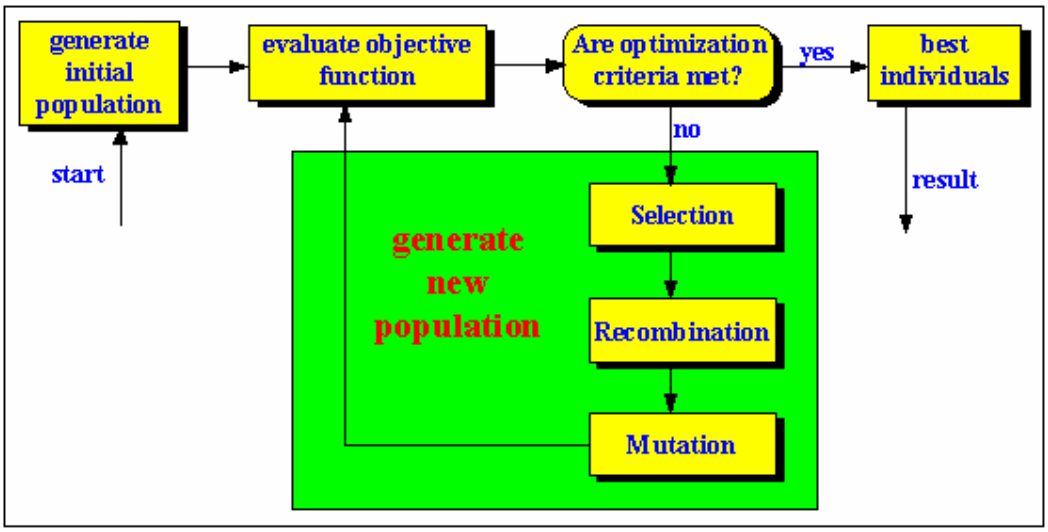

Figure 1: Schematizzazione di un algoritmo genetico standar

Ad ogni individuo all'interno della popolazione è assegnato un valore di fitness, che esprime la bontà della soluzione per il problema. La fitness determina quanto un individuo potrà propagare i suoi geni (il suo codice genetico) alle generazioni seguenti. Alle soluzioni migliori sono assegnati valori di fitness più alti rispetto alle soluzioni peggiori.

L'evoluzione è realizzata usando un insieme di operatori genetici, che manipolano il codice genetico. La maggior parte degli algoritmi genetici includono operatori che selezionano gli individui per la riproduzione, producono nuovi individui basandosi su quelli selezionati, e determinano la composizione della popolazione per la generazione seguente. Il *Crossover* e la *mutazione* sono due di questi operatori.

1.1.1 Funzione Fitness

Per ciascun problema da risolvere deve essere costruita una specifica funzione fitness. Dato un particolare cromosoma, la funzione fitness restituisce un singolo

valore numerico "fitness" o una "figura di merito", che si suppone sia proporzionale alla utilità o abilità dell'individuo che il cromosoma rappresenta.

La funzione fitness, insieme allo schema di codifica usata, è l'aspetto cruciale di ogni GA.

La stima di funzioni approssimate è una tecnica che può qualche volta essere usata se la funzione fitness è troppo complessa o lenta da valutare. Se può essere creata una funzione più veloce che approssimativamente dà il valore della funzione fitness "vera", il GA può trovare, in un dato tempo di CPU, un cromosoma migliore di quello che avrebbe trovato usando la vera funzione fitness.

1.1.1.1 Problemi di Fitness Range

All'inizio di un'esecuzione, i valori per ciascun gene dei diversi membri della popolazione sono casualmente distribuiti. Conseguentemente, c'è una grande propagazione di fitness individuali. Come l'algoritmo progredisce particolari valori per ogni gene cominciano a predominare. Mentre la popolazione converge, il range del fitness si riduce. La variazione nel range del fitness spesso porta a problemi di convergenza prematura o fine lenta.

1.1.2 Convergenza Prematura

Un classico problema con i GA è che i geni provenienti da pochi individui con un fitness comparabilmente alto (ma non ottimale) possono rapidamente dominare la popolazione, causando la convergenza a un massimo locale. Una volta che la popolazione converge, l'abilità del GA di continuare la ricerca per una soluzione

migliore è effettivamente eliminata: il crossover di individui quasi identici può portare ben pochi miglioramenti. Solo la mutazione rimane per poter esplorare nuove zone, e questo semplicemente porta a una ricerca lenta e casuale.

Per far lavorare bene il GA su una popolazione finita, dobbiamo modificare la maniera con cui vengono scelti gli individui per la riproduzione.

1.1.2.1 Fine lenta

Questo problema è opposto al precedente. Dopo molte generazioni, la popolazione sarà convergente, ma non avrà localizzato precisamente il massimo locale. Il fitness medio sarà alto, ma ci sarà poca differenza tra la media e il miglior individuo. Le stesse tecniche usate per combattere la convergenza prematura combattono anche questo problema.

1.1.3 Riproduzione

La *selezione* dei genitori ha il compito di allocare opportunità riproduttive a ciascun individuo. In principio gli individui migliori sono copiati dalla popolazione iniziale in una "piscina di accoppiamento" (*mating pool*), dove gli individui migliori hanno molta probabilità di essere copiati più volte, mentre i peggiori potrebbero non essere

copiati affatto. Dopo di ciò, coppie di individui vengono tirati fuori dalla piscina e fatti accoppiare. Questo viene ripetuto finché la piscina rimane vuota, la sua dimensione è pari a quella della popolazione. Il comportamento di un GA dipende molto da come gli individui vengono scelti per andare nel mating pool. Esistono diversi metodi, quello più usato è chiamato "Roulette wheel Selection". Gli individui sono selezionati in base al loro fitness: i migliori individui hanno maggiore probabilità di essere selezionati, si può immaginare una roulette dove vengono piazzati tutti i cromosomi, ognuno dei quali occupa uno spazio proporzionale al suo fitness. Poi "si lancia la pallina" e si seleziona il cromosoma.

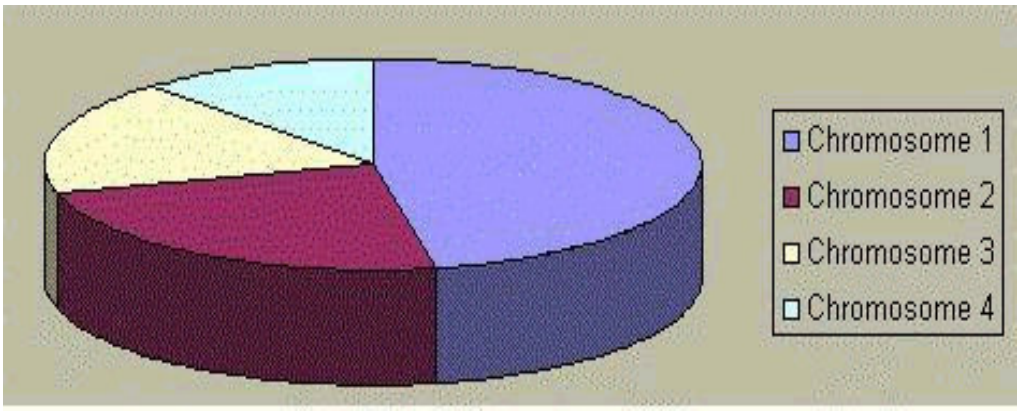

Figure 2: Schematizzazione dell' algoritmo di selezione

Il *crossover* prende due individui, e taglia le stringhe dei loro due cromosomi in qualche posizione scelta a caso, per produrre due segmenti "testa" (head)} e due segmenti "coda" (tail). I segmenti coda sono poi scambiati per produrre due nuovi cromosomi di lunghezza completa. Ciascuno dei figli eredita alcuni geni da ogni genitore. Questo è conosciuto come single point crossover.

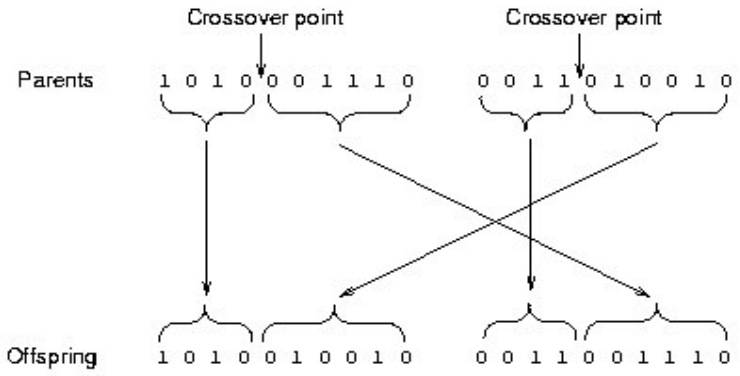

Figure 2: Single-point Crossover

Comunque, sono stati inventati molti diversi algoritmi di crossover, che spesso coinvolgono più di un punto di taglio.

La *mutazione* è applicata a ogni figlio singolarmente dopo il crossover. Viene alterato a caso ogni gene con una data probabilità. Gl'operatori base sono chiamati "Binary Mutation" e "Real Valued Mutation" rispettivamente. Il primo operatore, che può essere usato nel caso di una codifica genetica binaria, scambia il gene 1 in 0 o viceversa. Per quanto riguarda il secondo operatore il problema è dato dalla scelta del range di mutazione. Lo step ottimo dipende ovviamente dal problema che si affronta. In alcuni casi è consigliabile uno step di mutazione basso, in altri è preferibile uno più grande.

Un possibile algoritmo può essere il seguente:

- mutated variable = variable ± range*delta;(+ o – con eguale probabilità)
- range = 0.5* domain of variable;(intervallo di ricerca)
- delta = sum(a(i)*2^-i), dove a(i)=1 con probabilità 1/m, o a(i)=0; m=20

Con m=20, l'algoritmo riesce a trovare l'ottimo con una precisione di range*2^-19.

La teoria tradizionale ritiene che il crossover sia più importante della mutazione per quanto riguarda la rapidità nell'esplorare lo spazio di ricerca. La mutazione porta un po' di "casualità" nella ricerca e aiuta ad assicurarci che nessun punto nello spazio abbia probabilità nulla di essere esaminato, inoltre previene la deriva genetica ovvero il convergere dei membri della popolazione verso qualche punto dello spazio di

ricerca. Questo è dovuto al fatto che un gene predominante si può propagare a tutta la popolazione. Una volta che un gene converge in questa maniera il crossover non può introdurre nuovi valori. Da questo si evince che mentre la popolazione si avvicina alla convergenza la mutazione diventa più produttiva del crossover. In questa situazione sarà utilizzata la cosiddetta "Evoluzione Naive" (solo selezione e mutazione).

Dopo che i figli sono stati prodotti attraverso la selezione, il crossover e la mutazione degli individui della vecchia generazione, bisogna calcolare il loro fitness e reinserirli nella popolazione. A questo punto si hanno due possibilità:

- **Global reinsertion**:

  Esistono differenti schemi:

  - la vecchia popolazione viene sostituita integralmente (pure reinsertion).
  - vengono prodotti figli in numero inferiore ai genitori che vengono rimpiazzati in maniera uniforme e casuale (uniform reinsertion).

- vengono rimpiazzati i genitori peggiori (elitist reinsertion).
- viene generata una prole maggiore di quella richiesta e vengono reinseriti solo i migliori individui della prole (fitness-based reinsertion).

- **Local reinsertion**

Nella local selection gli individui vengono selezionati da un insieme limitato e contiguo. La reinserzione avviene esattamente nello stesso insieme, in questo modo viene preservata la località dell'informazione.

Per la selezione di un genitore da rimpiazzare e per quella di un figlio da inserire viene seguito uno di questi schemi:

- Tutti i figli vengono inseriti nell'insieme e gli individui vengono rimpiazzati in modo casuale,
- Tutti i figli vengono inseriti nell'insieme e gli individui peggiori vengono rimpiazzati.
- I figli migliori vanno a sostituire gli individui peggiori nell'insieme,
- I figli migliori prendono il posto dei genitori nell'insieme,
- I figli migliori vanno a sostituire individui scelti a caso nell'insieme,
- I genitori vengono rimpiazzati dai figli migliori.

Nell'implementazione si è utilizzato un metodo ibrido tra l'elitismo[1] e il fitness-based reinsertion che previene la perdita di informazione. Ad ogni generazione, un dato numero di genitori, quelli con basso fitness, è sostituito dallo stesso numero di individui della prole con alto fitness (vedi figura).

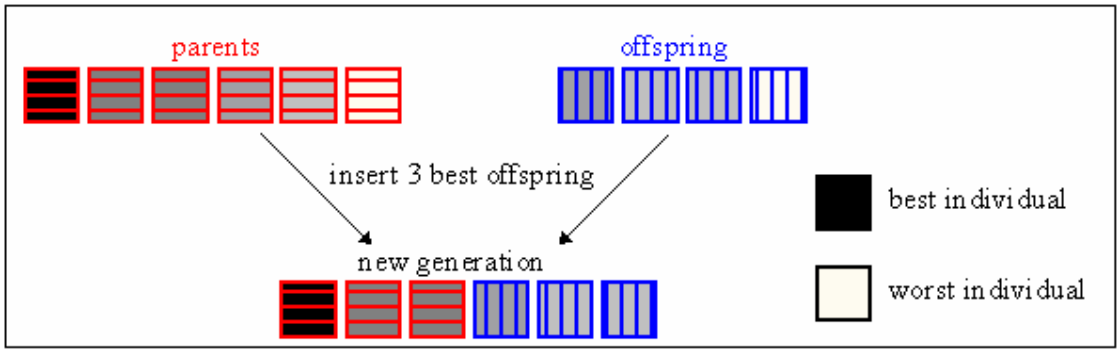

Figure 4: Schematizzazione dell'algoritmo di rimpiazzamento

1.1.4 Convergenza

Se il GA è correttamente implementato, la popolazione evolverà in molte generazioni in modo che il fitness del miglior individuo e la media in ogni generazione cresca verso l'ottimo globale. La convergenza è la progressione verso la crescente

[1] Quando creiamo una nuova popolazione con crossover e mutazione abbiamo una grande probabilità di perdere il miglior cromosoma. L'elitismo è un metodo che prima copia il miglior cromosoma (o i pochi migliori) nella nuova popolazione e il resto viene fatto in maniera classica. L'elitsmo può far crescere rapidamente le performance del GA perché evita la perdita della migliore soluzione trovata.

uniformità. Un gene converge quando il 95% della popolazione condivide lo stesso valore. La popolazione converge quando tutti i geni convergono.

1.1.5 Esplorazione e Sfruttamento (Exploration & Exploitation)

Un qualsiasi algoritmo di ottimizzazione efficiente, deve usare due tecniche per trovare il massimo globale: esplorazione (exploration) per esaminare nuove e sconosciute aree dello spazio di ricerca, e sfruttamento (exploitation) per fare uso dei punti precedentemente visitati per trovare punti migliori. Queste richieste sono contraddittorie, e un buon algoritmo di ricerca deve trovare un buon compromesso tra le due. Una ricerca puramente casuale è buona per l'esplorazione, ma non fa nessuno "sfruttamento", mentre un metodo puramente di scalata (sceglie sempre il cromosoma migliore per le nuove generazioni) è buono per lo sfruttamento, ma fa poca esplorazione. La combinazione di queste due tecniche può essere abbastanza efficace, ma è difficile sapere dove si trova l'equilibrio migliore (cioè quanto sfruttamento bisogna fare prima di arrendersi e esplorare di più?).

Un GA combina insieme esplorazione e sfruttamento allo stesso tempo e in modo ottimale; questo è vero da un punto di vista teorico, in pratica ci sono problemi inevitabili. Infatti per dimostrare quanto sopra sono state fatte alcune semplificazioni:

1. la popolazione è infinita

2. la funzione fitness riflette accuratamente l'utilità della soluzione

3. i geni in un cromosoma non interagiscono significativamente

In pratica la (1) non può mai essere soddisfatta, questo porta ad errori nella definizione del GA (i.e. deriva genetica).

1.2 Possibili applicazioni

L'applicazione dei GAs nel controllo può essere classificata in due aree distinte: off-line design e on-line optimisation. Le applicazioni off-line sono quelle più utilizzate e di maggior successo. Le applicazioni on-line sono usate di rado a causa delle difficoltà associate con l'uso dei GAs in tempo reale. I GAs sono stati applicati nei problemi di "controller design" e dei sistemi di identificazione. Altre applicazioni includono le "fault diagnosis", analisi di stabilità, piazzamento di sensori-attuatori, e altri problemi di carattere combinatorio.

1.2.1 Controllori

I GA sono stati largamente utilizzati per la realizzazione di controllori off-line. Principalmente sono stati utilizzati per ottenere parametri di controllo (per varie classi di controllori), o strutture di controllo ed in alcuni casi entrambi. Sono stati inoltre combinati con controllori fuzzy e controllori neurali per realizzare schemi di controllo intelligente.

1.2.1.1 Ottimizzazione di parametri

Nei primi anni 90 i GA vennero inizialmente studiati come mezzi alternativi per la messa a punto dei controllori PID (Proporzionali-Integrali-Derivativi). Nello studio di Oliveira e Sequeira del 1991 viene utilizzato un GA standard per determinare le stime iniziali dei valori dei parametri PID. Essi applicarono la loro metodologia a numerose classi di sistemi lineari tempo invarianti (LTI).

Wang e Kwok nel 1992 crearono un GA utilizzando "micro operatori" di inversione e preselezione per mettere a punto i controllori PID.

In uno studio indipendente Porter e Jones [1992] proposero una tecnica basata sul GA come metodo generico per la messa a punto dei controllori digitali PID.

Più recentemente Vlachos e altri [1999] hanno applicato i GA nella messa a punto di controllori PI (Proporzionali-Integrali) decentralizzati per processi multivariabili.

Nel 1997 Onnen ha applicato i GA per la determinazione di una sequenza di controllo ottimale in un MBPC (Model-Based Predictive Control).

Un ulteriore approccio al controller design utilizzando i GA consiste nell'applicare la metodologia indirettamente. In questo tipo di schema il GA manipola i parametri di ingresso in un processo di controller design stabilito che a sua volta produce il

controllore finale. Il metodo LQG (Linear Quadratic Gaussian)è stato utilizzato in questa maniera.

Nel disegno LQG un GA può essere utilizzato per la ricerca dei valori della matrice dei pesi in modo che soddisfi le specifiche di progetto, in quanto la selezione manuale degli elementi della matrice non è semplice.

1.2.1.2 Applicazione a controllori fuzzy/neurali

Le limitazioni dei controllori convenzionali per ciò che riguarda l'applicazione ai sistemi complessi e dinamici ha motivato ulteriori studi in quello che viene definito il campo dei sistemi di controllo intelligente. Le due tecniche più conosciute sono il controllo fuzzy, e il controllo neurale che è maggiormente indicato per i sistemi scarsamente modellati e non lineari. Linkens e Nyongesa hanno presentato, nel 1996, una valutazione di entrambe le tecniche.

I GA sono stati utilizzati nel tentativo di ottimizzare vari aspetti dei controllori intelligenti. Nel controllo fuzzy, un GA può essere utilizzato per creare una regola-base fuzzy e mettere appunto i parametri della funzione di membership associata. Mentre nel controllo neurale, un GA può funzionare come scelta alternativa per l'apprendimento dei valori di peso. E' inoltre stata aperta un diatriba sulla possibilità di utilizzare i GA per la ottimizzazione della tipologia di una rete neurale.

Sotto verranno illustrati alcuni esempi di applicazione di GA sul controllo intelligente.

Ichikawa e Sawa [1992] usarono una rete neurale come diretta sostituzione di un controllore convenzionale. I pesi furono ottenuti utilizzando un approccio GA. Ogni

elemento della popolazione rappresentava una distribuzione di peso per la rete e la struttura e la funzione di attivazione vennero stabilite a priori. Il beneficio di questo approccio consisteva nel fatto che non era necessario inserire modelli di insegnamento ne funzioni oggettive.

Il lavoro nel campo dei GA applicato al controllo fuzzy è genericamente diviso in due categorie: la realizzazione delle funzioni di appartenenza e la deduzione di regole base. Gli esperti del primo approccio tendono a sostenere che la forma delle regole dovrebbe essere conosciuta a priori, e che la maggiore incertezza risiede nello sviluppo delle funzioni di appartenenza associata. In oltre l'uso di una regola base statica riduce il livello necessario di complessità di calcolo, e questa può essere una ulteriore ragione per cui questo approccio viene preferito.

Linkens e Nyongesa nel 1995 presero in considerazione sia l'applicazione on–line che quella off-line dei GA per quel che riguarda il controllo fuzzy; analizzarono il processo completo del disegno fuzzy incluse la deduzione delle regole di controllo e l'ottimizzazione delle funzioni di appartenenza; in oltre, offrirono una discussione ponderata degli operatori e dei parametri dei GA dalla prospettiva del controllo fuzzy.

1.2.2 Identificazione

Molte applicazioni di controllo, come ad esempio, il controllo di previsione basato su un modello, richiedono un modello matematico del processo che si vuole controllare.
In alcuni casi, modelli accurati possono essere ottenuti analiticamente attraverso la considerazione dei processi fisici conosciuti. Questo approccio è generalmente appropriato per sistemi lineari, deterministici, tempo invarianti, e SISO (Single Input Single Output), per i quali è possibile avere una sufficiente conoscenza dei processi del sistema; tuttavia la maggioranza dei sistemi reali non coincide con questa categoria, infatti spesso sono non lineari e scarsamente conosciuti. Il black-box modelling, conosciuto comunemente come system identification, è spesso l'unico approccio realistico utilizzabile; in questo caso i dati input-output del sistema vengono utilizzati per generare una relazione matematica fra input ed output.
Il system identification può essere suddiviso in 2 problemi correlati fra loro:

- Selezione di una struttura di modello adeguata
- Stima dei parametri del modello.

Esistono tecniche altamente sviluppate per la stima dei parametri di modelli lineari e di modelli non lineari, ma lineari nei parametri; mentre sono sempre in fase di studio tecniche per la selezione di strutture e per la stima di parametri non lineari.

L'applicazione dei GA al system identification ha ricevuto un notevole interesse a partire dallo studio di Kristinsson e Dumont del 1992, nel quale applicarono i GA all'identificazione del sistema sia nei sistemi a tempo continuo sia in quelli a tempo discreto. Questo tipo di tecnica può essere utilizzata per le applicazioni on-line ed off-line. I GA furono utilizzati per identificare direttamente i poli e gli zeri, oppure per estrapolare i parametri fisici.

La funzione di costo utilizzata era rappresentata dall'errore fra l'output stimato e quello effettivo, in un ventaglio di dati composti dalla coppia di input-output attuale e i precedenti 30 campioni.

Per ogni punto campione, la popolazione venne sviluppata per ulteriori 3 generazioni. Kristinsson e Dumont ottennero risultati comparabili o addirittura migliori delle tecniche già conosciute, ma è necessario notare che questo metodo necessita di una spesa di calcolo decisamente elevata.

Successive applicazioni nel system idenification hanno cercato di ottimizzare i parametri o la struttura del modello, oppure entrambe simultaneamente.

1.2.3 Applicazioni on-line

Le applicazioni on-line presentano una sfida particolare per i GA; anche se, per il momento non sono stati ottenuti grandi successi in questo campo. I vantaggi dell'utilizzo di un GA per le applicazioni di control systems on-line sono gli stessi

che abbiamo già osservato nelle applicazioni off-line, tuttavia l'approccio ad un GA on-line deve essere affrontato con particolare cautela.

In una applicazione in tempo reale, c'è solo un limitato lasso di tempo fra due punti decisione nel quale un ottimizzatore può venire eseguito; considerando il potere di calcolo attuale, è improbabile che un GA possa eseguire la convergenza all'interno del tempo limite di campionatura di una tipica applicazione di controllo; ne deriva che solo un certo numero di generazioni può essere preso in considerazione. Se il GA si è evoluto solo per poche generazioni è chiaro che le performance della popolazione possono essere scarse. È possibile invece ottenere un accettabile livello di convergenza attraverso sistemi con tempi di campionatura più lunghi. Un ulteriore complicazione è rappresentata poi dal fatto che il sistema, visto dalla prospettiva dell'ottimizzatore, cambia nel tempo, così il segnale di controllo generato in un determinato momento può diventare completamente inappropriato il momento successivo.