UNIVERSITÀ DEGLI STUDI DI CATANIA
FACOLTÀ DI SCIENZE MATEMATICHE, FISICHE E NATURALI
CORSO DI LAUREA SPECIALISTICA IN INFORMATICA

---

EVA SCIACCA

# ALGORITMI EVOLUTIVI MULTIOBIETTIVO E *POSSIBILISTIC WORST-CASE DISTANCE* PER L'OTTIMIZZAZIONE DI CIRCUITI ELETTRONICI

---

TESI DI LAUREA

---

RELATORE:
Chiar.mo Prof. Angelo Marcello Anile

CORRELATORE:
Salvatore Spinella, Ph.D.

---

ANNO ACCADEMICO 2005 - 2006

# Indice

# Introduzione

In molti campi di ricerca e di sviluppo l'utilizzo di strumenti come i simulatori si rivela essere un aspetto fondamentale se non indispensabile per l'attività stessa. Nell'ambito della progettazione di circuiti elettronici tutto si basa sull'utilizzo di questi strumenti, che sono in grado di prevedere il funzionamento del sistema che si sta studiando, risolvendo complesse equazioni. L'accuratezza della risposta dipende principalmente dalla precisione dei modelli matematici implementati e dall'accuratezza della caratterizzazione dei parametri che appaiono in tali modelli. In questo ambito risulta fondamentale l'attività di caratterizzazione e modeling attraverso la quale determinare il miglior set di parametri in grado di descrivere il comportamento di tali sistemi. Tale caratterizzazione può essere fatta solo rispetto a intervalli di confidenza, e più in generale considerando un grado di incertezza intrinseco nell'individuazione di tali parametri.

La complessità di questi modelli comporta per il design microelettronico un tempo considerevole per il sizing dei circuiti analogici che devono soddisfare precisi requisiti di funzionamento. La ragione principale di tale complessità è dovuta al fatto che le relazioni tra le grandezze del dispositivo e le sue performance sono non lineari, e ciò, unito all'incertezza nella caratterizzazione dei modelli dei dispositivi, può rendere i risultati delle simulazioni poco significativi per il design in oggetto.

In questo lavoro di tesi vengono presentate metodologie per il sizing dei circuiti analogici alternative a quelle più diffuse nell'industria microelettronica. L' applicazione sulla quale sono state testate tali metodologie è stata il sizing dei dispositivi MOS di un two-stage OTA. In particolare la prima metodologia analizzata è quella che fa uso di algoritmi evolutivi per l'ottimizzazione multiobiettivo (Multi Objective Evolutionary Algorithm –MOEA–), mentre la seconda metodologia proposta utilizza i concetti di insieme fuzzy e teoria della possibilità per il calcolo e la minimizzazione della Worst-Case Distance. A quest'ultima metodologia è stato dato il nome di Possibilistic Worst-Case Distance –PWCD–.

Nel **Capitolo 1** viene presentato il caso di studio preso in considerazione per valutare le prestazioni di ottimizzazione circuitale degli algoritmi evolutivi multiobiettivo e della nuova metodologia proposta di PWCD. In questa applicazione test è stato effettuato il sizing di dispositivi MOS di una rete circuitale che rappresenta un Operational Transconductance Amplifier a due stadi (two-stage OTA), del quale vengono descritte le principali metriche di funzionamento.

Attualmente l'industria microelettronica utilizza per il design metodi di ottimizzazione mono obiettivo, ed i tentativi di ottimizzazione multiobiettivo sono svolti ancora in ottica esplorativa. E' stato proposto un approccio multiobiettivo come alternativa all'approccio della funzione di costo, allo scopo di aumentare l'efficienza della progettazione dei dispositivi nei circuiti analogici.

Nel **Capitolo 2** si discute della Ottimizzazione Multiobiettivo e degli algoritmi genetici destinati a tale tipologia di ottimizzazione.

Il **Capitolo 3** descrive in dettaglio gli algoritmi genetici utilizzati nella ottimizzazione del problema test: NSGA2 e SPEA2. In particolare è stata focalizzata l'attenzione sullo studio dell'algoritmo MultiOb sviluppato dal Dipartimento di Ottimizzazione dell' Istituto ITWM Fraunhofer per la Matematica Industriale [14]. Inoltre, viene descritta la formulazione del problema e vengono valutati i risultati. Obiettivo di tale applicazione test è stato quello di ottenere una classe di dispositivi ottimizzati che coprano i range di performance ottimali rispetto ai trade-off di queste ultime.

I capitoli seguenti introducono la metodologia di Possibilistic Worst-Case Distance.

Il **Capitolo 4** approfondisce il concetto di Worst-Case Analysis che mira ad identificare il caso peggiore e quindi fa in modo di verificare le performance soddisfacenti.

Il **Capitolo 5** presenta brevemente la teoria degli insiemi fuzzy. Inoltre approfondisce la teoria della possibilità.

Infine, il **Capitolo 6** descrive la metodologia di PWCD e la sua implementazione nel caso specifico di utilizzo nell'applicazione test di sizing circuitale. Tale metodologia è risultata essere innovativa perchè, prendendo in considerazione l'incertezza di alcuni parametri circuitali, permette una progettazione robusta in termini di accettabilità nelle sue performance poichè tutte le specifiche vengono rispettate.

# Capitolo 1

# Applicazione Test : Two-Stage OTA

All'interno del design microelettronico viene speso un tempo considerevole nel sizing dei circuiti analogici per soddisfare precisi requisiti di funzionamento. La ragione principale di questo è dovuta a determinate relazioni non lineari tra le grandezze del dispositivo e i suoi funzionamenti [15, 17].

In questa applicazione test, il processo di ottimizzazione viene accoppiato all'utilizzo di un simulatore circuitale (Spice) in modo tale da valutare le performance del circuito. Le versioni dei simulatori Spice utilizzate sono state ngspice e PSpice, le quali implementano modelli MOSFET BiSim3 per le caratterizzazioni I-V. La temperatura per le simulazioni è stata posta a 27 $^{\circ}$ C. Mentre i parametri delle model card per i transistor PMOS ed NMOS sono stati i parametri di default.

In questa applicazione test è stato effettuato il sizing di dispositivi MOS di una rete di circuito rappresentante un Operational Transconductance Amplifier a due stadi (two-stage OTA).

L' Operational Transconductance Amplifier è un amplificatore in cui tensioni in input differenziali producono correnti in uscita; per tale motivo è un dispositivo VCCS (voltage controlled current source). L'OTA è simile ad un comune amplificatore operazionale : l'impedenza di ingresso presenta un valore molto elevato, teoricamente infinito, mentre l'impedenza di uscita ha valore basso, idealmente nullo.

L'OTA è un dispositivo utile singolarmente ma alcune volte è utilizzato con altri dispositivi per realizzare filtri, comparatori, generatori di onde o convertitori.

## 1.1 Panoramica sui dispositivi analogici

Oggi esistono diversi processi tecnologici per realizzare i circuiti integrati. Il più popolare di questi è il CMOS (Complementary Metal Oxide Semiconductor).

Nel processo CMOS, sia il MOSFET di canale-n (NMOS) che di canale-p (PMOS), vengono utilizzati per realizzare le funzioni logiche. Dal momento che il transistor PMOS è in "on" quando un basso livello di voltaggio (zero) viene applicato alla porta, e il transistor NMOS è "on" quando il potenziale di porta è alto (uno) questi dispositivi possono essere utilizzati per formare circuiti complementari dove il consumo di potenza statico è piccolo. L'area richiesta per implementare una specifica funzione è anche piccola, cioè l'insieme di funzionalità che possono essere implementate su di un chip è ampio.

Il basso consumo di potenza e il fatto che il costo del processo manifatturiero di circuiti CMOS è basso in relazione ad esempio ai circuiti bipolari, ha reso il CMOS il processo tecnologico più popolare per i circuiti digitali.

Sebbene i transistor bipolari possano lavorare ad alte frequenze, le proprietà discusse sopra e il fatto che i circuiti analogici e digitali sono utilizzati sullo stesso chip ha reso il processo CMOS ancora più importante per il design analogico.

Processi speciali, come BiCMOS, combinano la tecnologia bipolare con la tecnologia CMOS. In questo modo il progettista può utilizzare dispositivi bipolari per funzionamenti critici per le parti analogiche e i CMOS per le parti digitali sullo stesso chip.

### 1.1.1 I dispositivi CMOS

In questa sezione viene discusso un semplice modello per un transistor CMOS. Sebbene questo modello non predice accuratamente il comportamento del transistor sub-micron, le relazioni sono ancora valutabili per capire la relazione tra differenti proprietà dei dispositivi. Inoltre, alcuni dei concetti qui presentati possono essere utilizzati anche in modelli più accurati.

Le equazioni presentate sono quelle normalmente utilizzate quando si eseguono calcoli manuali. Questi sono anche una parte integrante dell'approccio di device sizing.

### Modelli a grande segnale

Le notazioni simboliche che rappresentano transistori PMOS ed NMOS sono mostrati in figura 1.1. Tali figure rappresentano le versioni simboliche a 4 terminali del transistor. In questo semplice modello si assume che i transistori abbiano 3 regioni operative.

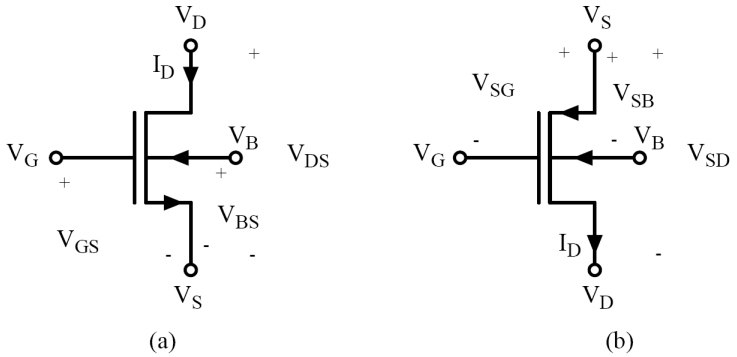



Figura 1.1: Definizioni di tensioni e correnti per transistori a 4-terminali NMOS (a) e PMOS (b).

Qui vengono date le equazioni per transistor NMOS. In ogni caso le relazioni e le equazioni per transistor PMOS sono simili. Per transistor NMOS le regioni di taglio, lineare e saturazione, sono definite dalle seguenti relazioni:

***taglio*** $V_{DS} < V_{GS} - V_{TH}$

***lineare*** $V_{DS} \leqslant V_{GS} - V_{TH}$ e $V_{GS} > V_{TH}$

***saturazione*** $V_{DS} \geqslant V_{GS} - V_{TH}$ e $V_{GS} > V_{TH}$

Qui $V_{TH}$, la tensione di threshold, è la minima porta al canale del voltaggio richiesta in maniera da creare un canale di conduzione tra drain e source nel transistor.

La tensione di threshold è modellata da:

$$V_{TH} = V_{TH0} + \gamma(\sqrt{2\phi_F - V_{BS}} - \sqrt{2\phi_F})$$

dove $V_{TH0}$ è la tensione di threshold quando $V_{BS}$ è zero. I termini addizionali sono risultato dell'effetto body. L'effetto body è un aumento nella tensione di threshold dovuta alla differenza di tensione tra il substrato (bulk) e il source del transistor.

Nella espressione sopra, $\phi_F$ è la differenza tra il potenziale di Fermi alla porta e il potenziale di Fermi nel substrato.$\gamma$ è la costante di body determinata dal processo dei parametri.

La corrente attraverso il transistor , $I_D$, dipende dalla regione operativa, essa è data da:

***taglio*** $I_D \approx 0$

***lineare*** $I_D = \mu_n C_{ox} \frac{W}{L} \left( (V_{GS} - V_{TH}) V_{DS} - \frac{V_{DS}^2}{2} \right)$

***saturazione*** $I_D = \frac{\mu_n C_{ox}}{2} \frac{W}{L} (V_{GS} - V_{TH})^2 (1 + \lambda V_{DS})$

dove $\mu_n$ è la mobilità degli elettroni vicino la superficie del silicio per il transistor NMOS, $C_{ox}$ è la capacitanza di porta per unità di area, $\lambda$ è la costante di impedenza risultante, e W ed L sono la larghezza e la lunghezza di canale del transistor.

Negli amplificatori analogici i transistor sono di solito polarizzati per lavorare nella regione di saturazione. La motivazione per questo è che quando un transistor opera in tale regione, la corrente è controllata principalmente dalla tensione del source. Comunque, un $\lambda$ non nullo introdurrà ugualmente una dipendenza non voluta sulla tensione drain-source.

Il concetto, utilizzato in modelli con alta accuratezza come BSIM3v3, EKV o Phillips MOS level 9, è simile ai modelli semplificati mostrati qui. Comunque, le espressioni per computare, ad esempio la corrente di drain del transistor, includono molti più effetti. Dunque, le espressioni utilizzate all'interno di questi modelli non sono indicate per calcoli manuali.

### Modelli a piccolo-segnale

Per variazioni a largo segnale il modello a largo segnale può essere utilizzato per calcolare il comportamento del circuito. In ogni caso, se vengono considerate solo le variazioni a piccolo segnale, il modello non lineare a piccolo segnale può essere linearizzato. Dunque, il modello a piccolo segnale risulta essere una linearizzazione del modello a grande segnale intorno al punto operativo del circuito.

Questo modello è valido soltanto per piccole perturbazioni del punto operativo. I modelli possono essere utilizzati in modo da semplificare i calcoli, ad esempio, delle proprietà di dipendenza dalla frequenza del circuito.

Un semplice modello a piccolo-segnale per il transistor NMOS è mostrato in figura 1.2. Qui $g_m$ è la transconduttanza e $g_{ds}$ è la conduttanza in uscita. I parametri a piccolo segnale sono estratti utilizzando le seguenti espressioni:

$$g_{ds} = \frac{\partial I_D}{\partial V_{DS}} = \frac{1}{2} \mu_n C_{ox} \frac{W}{L} (V_{GS} - V_{TH})^2 \lambda$$

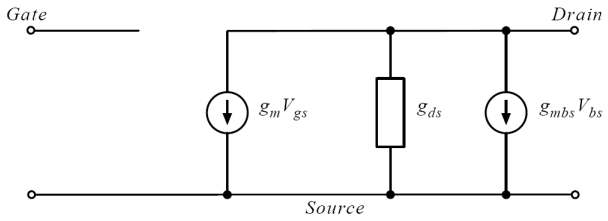



Figura 1.2: Semplici modelli a piccolo segnale di un transistor MOS.

$$g_m = \frac{\partial I_D}{\partial V_{GS}} = \mu_n C_{ox} \frac{W}{L}(V_{GS} - V_{TH})$$

$$g_{mbs} = \frac{\partial I_D}{\partial V_{BS}} \frac{\partial V_{TH}}{\partial V_{BS}} = g_m \frac{\lambda}{2\sqrt{2\phi_F + V_{BS}}} = \eta g_m$$

Come nel caso del modello a grande segnale, i più accurati modelli di transistor a piccolo segnale tengono in conto diversi parametri in più.

### 1.1.2 Metriche di funzionamento per gli amplificatori analogici

Ci sono diverse importanti metriche di funzionamento che devono essere prese in considerazione quando si progetta un amplificatore analogico. L'importanza di tali metriche sono, certamente, l'alta dipendenza su quale applicazione si intende che l'amplificatore venga utilizzato.

In questa sezione verranno discusse alcune più comuni metriche di funzionamento per gli amplificatori analogici. A seconda di quale sarà l'applicazione dell'amplificatore vengono scelte altre metriche che possono essere ugualmente interessanti. Inoltre, bisogna sottolineare il fatto che per alcune metriche di funzionamento esistono certamente delle definizioni alternative.

La funzione di trasferimento più generale possibile per un amplificatore è la seguente:

$$A(s) = \frac{A_0 \left(1 + \frac{s}{z_1}\right) ... \left(1 + \frac{s}{z_n}\right)}{\left(1 + \frac{s}{p_1}\right) ... \left(1 + \frac{s}{p_m}\right)}$$

Qui $z$ denota uno zero e $p$ denota un polo. $A0$ è il guadagno in DC, cioè, il guadagno quando $s$ è uguale a zero.

Inoltre, il segnale di uscita per un amplificatore differenziale può essere scritto come:

$$V_{out}(s) = A_{CM}(s) \left[\frac{V_p(s) + V_n(s)}{2}\right] + A_{DM}(s) \left[V_p(s) - V_n(s)\right]$$

Dove $A_{CM}(s)$è il guadagno in modalità comune e $A_{DM}(s)$ è il guadagno differenziale. $V_p(s)$ è il segnale di input positivo, mentre $V_n(s)$ è il segnale di input negativo.

### Larghezza di banda e frequenza all'unità di guadagno

In figura 1.3 è mostrata la risposta di magnitudo e di fase per un amplificatore a due poli. La larghezza di banda di un amplificatore è definita come la banda di frequenza per la quale l'amplitude è all'interno dei 3 dB del suo massimo valore. La larghezza di banda per l'amplificatore è in figura 1.3 di circa 100 rad/s.

Un'altra metrica di funzionamento è la larghezza di banda all'unità di guadagno (UGBW). L'UGBW è definita come il range di frequenza per il quale l'amplificatore ha un guadagno che è circa uguale all'unità.

Inoltre, alla frequenza all'unità di guadagno, $\omega_u$, l'amplificazione del segnale di input è pari all'unità, cioè, sotto questa frequenza il segnale di input non viene più amplificato.

La frequenza all'unità di guadagno è definita come:

$$|A(j\omega_u)| = 1$$

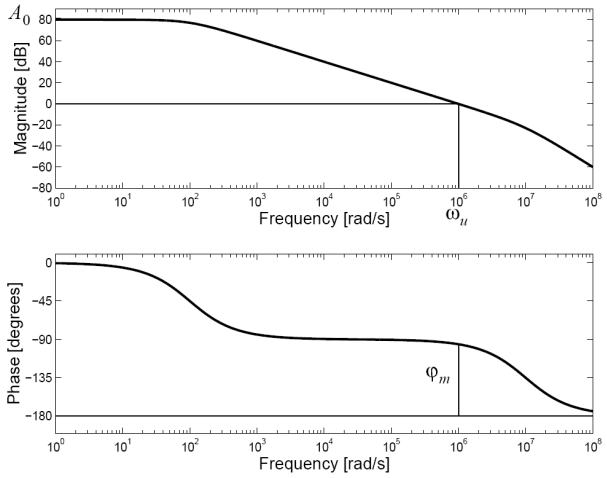



Figura 1.3: Risposte di Magnitudo e Fase di un amplificatore.

### Margine di fase

Il margine di fase,$\varphi_m$, è definito come:

$$\varphi_m = \arg(\beta A(j\omega_u)) - (-\pi)$$

Qui $\beta$ è il fattore di feedback in una configurazione di un amplificatore a ciclo chiuso. Il margine di fase è un misura di stabilità per l'amplificatore.

### PSRR positivo e negativo

Rapporto di reiezione della potenza fornita (Power Supply Rejection Ratio - PSRR) misura l'abilità degli amplificatori di sopprimere le variazioni nella potenza fornita dalle tensioni.

In un caso ideale, una modifica nella tensione fornita, non apporterebbe alcun effetto al funzionamento dell'amplificatore. Comunque, nella realtà, cambiando la potenza fornita dalla tensione si avrebbero delle conseguenze sui livelli di polarizzazione e quindi sull'operatività del circuito. Se è presente una larga variazione nella potenza fornita dalla tensione dovuta ad esempio ad alte attività di switching nel circostante insieme di circuiti digitali, è importante che queste variazioni abbiano un piccolo impatto sul funzionamento dell'amplificatore.

Sono di interesse di studio sia il PSRR positivo, cioè la soppressione di variazione nella potenza positiva fornita dalla tensione, che il PSRR negativo, cioè la soppressione di variazione nella potenza negativa fornita dalla tensione.

Le definizioni sono:

$$PSRRp(j\omega) = 20\log\left(\frac{A_{DM}(j\omega)}{A_{VDD}(j\omega)}\right)$$

$$PSRRn(j\omega) = 20\log\left(\frac{A_{DM}(j\omega)}{A_{VSS}(j\omega)}\right)$$

Dove $A_{VDD}$ è la magnitudo della risposta in frequenza dalla potenza positiva fornita al terminale di uscita e $A_{VSS}$ è la magnitudo della risposta in frequenza dalla potenza negativa fornita al terminale di uscita.

### Rapporto di reiezione di modo comune

Il Rapporto di reiezione di modo comune (Common Mode Rejection Ratio) è una misura di come vengono soppressi i segnali di modo comune non voluti sul terminale di ingresso dell'amplificatore. Nella

$$CMRR(j\omega) = 20\log\left(\frac{A_{DM}(j\omega)}{A_{CM}(j\omega)}\right)$$

Il guadagno del segnale di modo comune è comparato al guadagno del segnale differenziale. Nel caso ideale il CMRR è infinitamente grande, cioè, il segnale di modo comune è non amplificato del tutto.

### Slew Rate

Con il termine slew rate si intende la massima pendenza ottenibile sul segnale in uscita.

$$SR = \max\left[\frac{dV_{out}}{dt}\right]$$

Questa è misurata applicando un largo gradino di tensione al terminale di input e misurando la pendenza al terminale di output. Quando occorre una limitazione sullo slew-rate l'amplificatore non si comporterà come un sistema lineare. Per esempio se viene applicato un grande gradino di tensione al terminale di input dell'amplificatore, il segnale in uscita è un gradino lineare con una costante di pendenza (appunto SR).

### Distorsione

Quando ad esempio un segnale sinusoidale è utilizzato come input per l'amplificatore, il segnale in uscita, nel caso ideale, è un'amplificazione lineare di tale segnale. Ma, nella realtà ciò non avviene. Infatti, la forma del segnale in uscita devia da quello che è il segnale in ingresso. Questo fatto è dovuto alla non linearità dell'amplificatore che altera il segnale.

Ci sono diversi tipi di non linearità negli amplificatori analogici. Prima di tutto gli elementi del circuito : transistori, capacitori, e resistori sono elementi non lineari. Secondariamente, altri effetti come le limitazioni sullo slew-rate causano un comportamento non lineare.

La distorsione appare come una componente della frequenza inattesa sull'output dei circuiti. Un modo per misurare la distorsione armonica, cioè l'effetto delle armoniche di alto ordine, è quello di misurare la distorsione armonica totale (Total Harmonic Distortion –THD– ) THD mette insieme le armoniche fondamentali e quelle di alto ordine. Viene calcolata sommando le armoniche di alto ordine paragonandole con quelle fondamentali come nella seguente relazione:

$$THD = 10\log\frac{\sum_{n=2}^{\infty} P_n}{P_{fundamental}}$$

dove$P_n$ è la n-esima armonica mentre $P_{fundamental}$ è la potenza della fondamentale.

Altre metriche di funzionamento per la linearità sono le distorsioni armoniche, punti di compressione ed intercettazione, e distorsione inter modulare. Il concetto di punti di intersezione è illustrato in figura 1.4.

I punti di intersezione sono calcolati estrapolando le curve dell'armonica fondamentale, seconda e terza come in figura 1.4 . Poiché la seconda armonica cresce quadraticamente, intersecherà la fondamentale in qualche punto , diciamo $IIP_2$.

Più in alto l'intersezione si viene a trovare, migliore è la soppressione dell'armonica. $IIP_3$ è calcolato in maniera molto simile. In questa maniera è anche possibile calcolare i termini di distorsione di alto ordine.

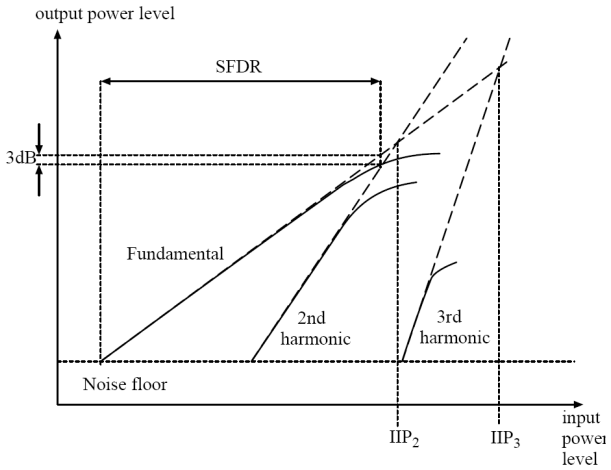



Figura 1.4: Punti di intersezione ed SFDR

**Rumore**

Nei MOSFET ci sono principalmente due tipi di sorgente di rumore, termico e flicker (rumore 1/f).

Il rumore termico si presenta nei resistori ed è il risultato dei movimenti casuali degli elettroni dovuti all'effetto termico. Il rumore flicker si crede sia il risultato di cariche "intrappolate" nel dispositivo per diverse ragioni e solo dopo rilasciate. A differenza del rumore termico, il rumore flicker dipende dalla frequenza in maniera inversamente proporzionale.

La densità spettrale del rumore all'uscita del circuito viene calcolata sommando il contributo di tutte le sorgenti di rumore indipendenti nel circuito, in questo modo:

$$S_{out}(\omega) = \sum_{n=1}^{N} S_n \left|H_n(\omega)\right|^2$$

Qui $N$ è il numero delle sorgenti di rumore, $S_i$ è la densità spettrale della sorgente di rumore, e $H_n$ è la risposta in magnitudo dalla sorgente di rumore all'output del circuito. Integrando il rumore su tutta la banda di frequenza si ottiene la potenza del rumore.

Una metrica di funzionamento comune per il rumore è il segnale al rapporto di rumore (Signal-to-Noise-Ratio – SNR–) definito da:

$$SNR = 10\log\left(\frac{P_{signal}}{P_{noise}}\right)$$

Dove il $P_{signal}$ è la potenza del segnale mentre$P_{noise}$ è la potenza del rumore. SNR viene definito per uno specifico range di frequenza sul quale vengono integrate le corrispondenti potenze di segnale e del rumore.

Altre misure includono l'effetto combinato di rumore e distorsione. SNDR (Signal-to-Noise plus Distortion Ratio) misura la degradazione dovuta all'effetto combinato del rumore e della distorsione all'interno della banda di frequenza. SNDR è definito come il rapporto della potenza di segnale dell'armonica fondamentale sulla somma della potenza di tutte le armoniche (quelle aliased e quelle dovute al rumore).

$$SNR = 10\log\left(\frac{P_{signal}}{P_{noise} + \sum_{n=2}^{\infty} P_n}\right)$$

Un'altra metrica, indicata in figura 1.4 e SFDR (Spurious-Free Dynamic Range). SFDR è definito come il range dinamico dove il segnale è (equamente) non affetto da rumore e distorsione in una specifica banda di frequenza. Mentre il piano del rumore impone un limite inferiore sulla potenza di segnale, la distorsione ne implica un limite superiore.

**Range di modo comune e range di uscita.**

Il range di modo comune (Common-Mode Range – CMD –) è definito come range a livelli di modo comune al quale il circuito risponde correttamente a segnali di input differenziali. Al fine di determinare il CMR, dovrebbe essere definito un livello accettabile di distorsione all'output del circuito.

Il range di uscita (Output Range – OR–) è definito come il range nel quale il segnale di uscita può variare, dato che la distorsione deve essere ad di sotto di un certo valore di soglia.

**Settling Time**

Il settling time denota il tempo richiesto per il segnale di uscita di un amplificatore per rientrare in una fascia assegnata quando viene applicato un segnale di ingresso a gradino. A seconda

della magnitudo del gradino, il settling può essere lineare o non lineare. Per un piccolo gradino, solo la larghezza di banda dell'amplificatore limita il settling time. In questo caso il settling è lineare. Il settling lineare determina un limite superiore su tutto il settling time. Invece, quando un largo gradino è applicato al terminale di ingresso, l'amplificatore incontra limitazioni dello slew rate dovute alla corrente finita che può essere fornita ai nodi capacitivi. In questo caso il settling è non lineare.

Il settling è calcolato applicando un gradino al terminale di ingresso e misurando il tempo fino a quando il segnale di output si trova ad un certo range del suo valore finale come mostrato in figura 1.5 . Il range esatto può variare a seconda dell'applicazione dell'amplificatore.

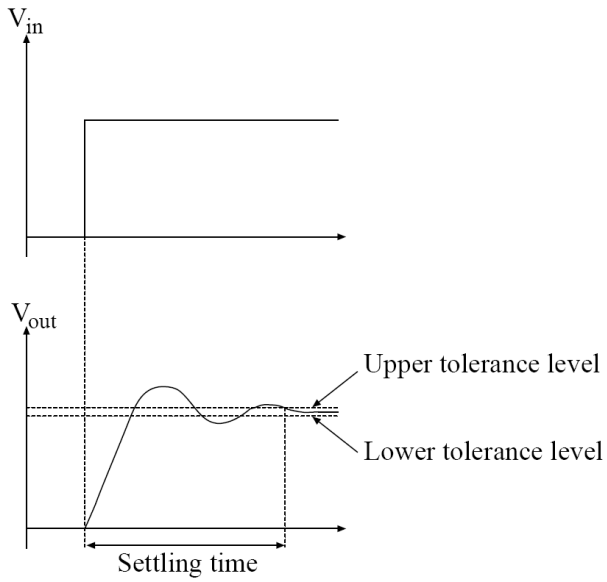



Figura 1.5: Misure del settling time dalla risposta del circuito al gradino di ingresso.

**Area del circuito, Consumo di potenza e Yield**

Alcune metriche di funzionamento non sono direttamente affette dalla funzionalità del circuito ma sono piuttosto interessanti poiché sono associate con il suo costo. Un esempio tipico di tale metrica di funzionamento è l'area di silicio occupata dal circuito. L'area di chip direttamente incide sui costi di manifattura ed è quindi importante rendere il circuito il più piccolo possibile.

Il consumo di potenza è oggi molto importante per quanto riguarda il largo numero di applicazioni che funzionano a batteria. Il consumo di potenza influisce direttamente sul tempo operativo di questi prodotti ed è quindi una importante metrica di funzionamento.

Durante il processo manifatturiero, una certa percentuale di tutti i circuiti non opera come previsto dalle specifiche. Ciò è dovuto a varie ragioni. Lo Yield (resa) è definito come il numero dei circuiti, presi da un insieme, che opera secondo le specifiche. Massimizzare lo Yield influisce direttamente sul costo della manifattura di un chip. Progettare in modo da aumentare lo Yield è quindi importante.

Inoltre, lo Yield è in parte dipendente dall'area del chip poiché il numero di errori di processo è approssimativamente lo stesso per unità di area. Così, una bassa percentuale di chip sarà affetta da questi errori se ciascun chip viene reso più piccolo.

## 1.2 Design Case

In questa sezione viene presentato il caso di studio preso in considerazione per valutare le prestazioni di ottimizzazione circuitale degli algoritmi evolutivi multiobiettivo (MOEA) e della nuova metodologia proposta di Possibilistic Worst-Case Distance (PWCD).

In questa applicazione test è stato effettuato il sizing di dispositivi MOS di una rete di circuito rappresentante un Operational Transconductance Amplifier a due stadi (two-stage OTA). La figura 1.6 rappresenta la topologia di tale circuito.

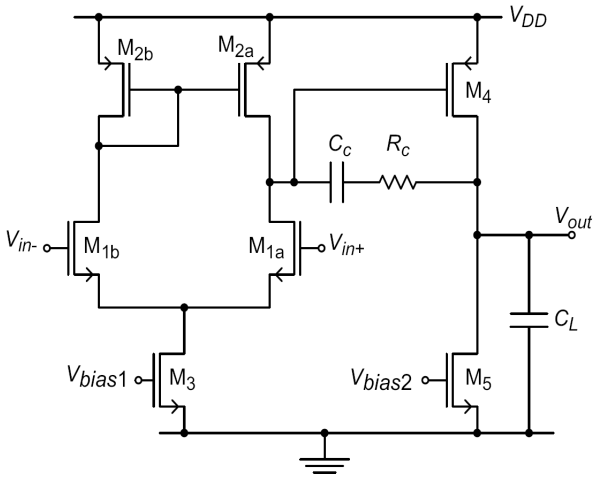



Figura 1.6: Topologia del circuito di un OTA a due porte.

I parametri da ottimizzare e i loro relativi range di definizione, così come le performance considerate nell'ottimizzazione variano nelle due applicazioni di MOEA e di PWCD e verranno descritti nei capitoli seguenti.

In questa applicazione test, il processo di ottimizzazione viene accoppiato all'utilizzo di un simulatore circuitale (Spice) in modo tale da valutare le performance del circuito. Le versioni dei simulatori Spice utilizzate sono state ngspice [1] e OrCad PSpice, le quali implementano modelli MOSFET BiSim3 per le caratterizzazioni I-V [8, 19].

### 1.2.1 Simulatore circuitale: Spice

I programmi di simulazione circuitale costituiscono uno strumento di estrema utilità per chi si occupa di progettazione di circuiti elettronici. Il loro impiego diviene pressoché indispensabile nel caso di progettazione di circuiti integrati. Infatti, sebbene il modo più semplice per verificare la rispondenza alle specifiche di progetto di un circuito elettronico sia quello di costruirne un prototipo ed effettuare misure su di esso, l'avvento della tecnologia dei circuiti integrati ha reso molto problematico questo procedimento. La realizzazione di un prototipo di circuito integrato deve essere effettuata con le stesse tecnologie impiegate per la realizzazione del circuito definitivo, con costi e tempi notevoli per una semplice verifica di progetto.

Per questo motivo, già a partire dai primi anni '70, sono stati sviluppati programmi di simulazione circuitale sempre più sofisticati e potenti. Lo scopo di questi programmi è, infatti, quello di costituire un'alternativa alla realizzazione di prototipi. In termini molto semplici, questi programmi simulano un banco di laboratorio sul quale sia possibile costruire un circuito elettronico e provarlo.

Da diversi anni uno dei programmi più accreditati nel campo della simulazione circuitale è SPICE (*Simulation Program with Integrated Circuit Enphasis*). La prima versione di questo programma fu sviluppata, durante la prima metà degli anni '70, all'Università di Berkeley in California. Negli anni successivi il programma ha subito continui miglioramenti e, accanto alla versione originale, sono state sviluppate versioni commerciali, prodotte da varie ditte. Attualmente sono disponibili un gran numero di prodotti che possono essere considerati come parte della famiglia SPICE. Questi programmi esistono sia per grandi calcolatori, che per Workstation. Da circa 10 anni sono anche disponibili versioni di SPICE per personal computer. Ovviamente, al variare del computer utilizzato, cambiano le prestazioni ottenibili dalle varie versioni in termini di tempo di calcolo e dimensioni del circuito che è possibile simulare. Tuttavia, tutti i programmi della famiglia di SPICE utilizzano, essenzialmente, gli stessi algoritmi di calcolo e richiedono le stesse modalità d'uso da parte dell'utilizzatore.

Il programma di gran lunga più diffuso a livello di personal computer è *Pspice*, prodotto dalla ditta *MicroSim Corporation* (ora incorporata nella ditta *OrCad*).

Le operazione che si possono eseguire con il programma PSpice possono essere così sommariamente riassunte:

- Analisi in corrente continua (DC): analisi in regime stazionario
  - calcola il punto di lavoro o di riposo del circuito con tutti i condensatori disconnessi (cioè aperti) e tutti gli induttori cortocircuitati.
  - se una o più delle caratteristiche tensione-corrente dei vari componenti utilizzati sono non lineari, come nel caso dei dispositivi a semiconduttori, le equazioni risolventi sono anch'esse non lineari ed il programma, per la soluzione, utilizza un metodo iterativo.
- Analisi in transitorio (TRAN): calcola correnti e tensioni rispetto al tempo
  - calcola le tensioni di nodo come forma d'onda in funzione del tempo ed essendo un'analisi per grandi segnali, non viene posto alcun limite all'ampiezza delle tensioni o delle correnti di ingresso.
- Analisi AC: linearizza il circuito nell'intorno del punto di lavoro e calcola l'uscita in funzione della frequenza
  - calcola tutte le tensioni di nodo di un circuito lineare nel formato di numeri complessi, funzione della frequenza imposta dal generatore sinusoidale applicato in ingresso.
  - nel caso dei circuiti non lineari, la soluzione è determinata facendo uso dell'analisi per piccoli segnali.
- Analisi in temperatura: analisi per diverse temperature
- Analisi del rumore: calcola il contributo di rumore di ogni componente e gli effetti sull'uscita
- Analisi di sensibilità: indica quali componenti sono piu'critici rispetto alla performance del circuito

- Analisi di Fourier: calcola i coefficienti dello sviluppo in serie di Fourier per le tensioni e le correnti
- Analisi Montecarlo: simulazioni multiple per certi tipi di analisi usando distribuzioni statistiche per i valori di alcuni componenti
- Analisi di distorsione: calcola caratteristiche di distorsione armonica utilizzando analisi AC ai piccoli segnali

La figura 1.7 mostra schematicamente il principio di funzionamento di SPICE.

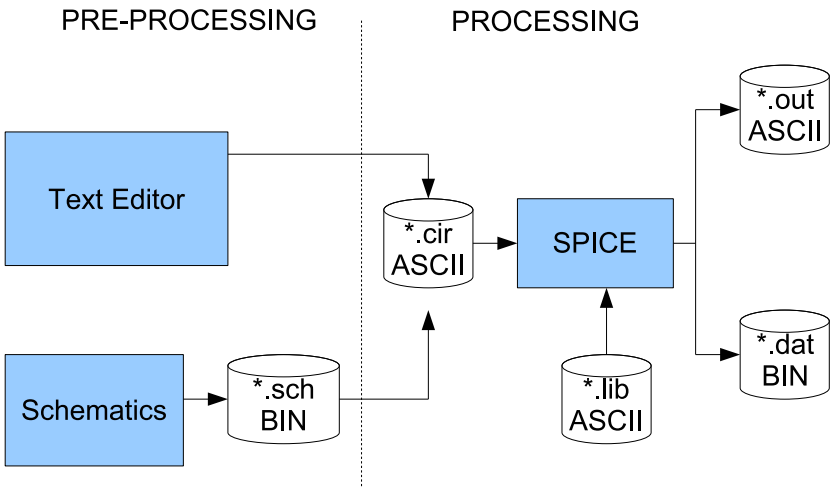



Figura 1.7: Schema di funzionamento di Spice.

# Capitolo 2

# Ottimizzazione Multiobiettivo e Algoritmi Evolutivi

Nella maggior parte dei problemi del mondo reale risulta necessario trovare una soluzione che ottimizzi contemporaneamente diversi obiettivi, i quali spesso si trovano in competizione tra loro. In questo caso il problema da affrontare non si riduce semplicemente alla ricerca di un massimo o di un minimo locale (o globale) per una data funzione: in un problema di Ottimizzazione Multiobiettivo si avverte l'esigenza di trovare una soluzione che sia ottimale contemporaneamente per tutte le funzioni obiettivo che descrivono il problema.

In genere, quando si affronta un problema di Ottimizzazione Multiobiettivo, la soluzione ottimale ottenuta non è mai unica (come nel caso di ottimizzazione ad un singolo obiettivo), ma ci si trova di fronte ad un insieme di soluzioni ugualmente ottimali rispetto al problema dato. Tale insieme di soluzioni prende il nome di Insieme Pareto Ottimale. Spetta poi alla figura del Decision Maker il compito di decidere quale delle soluzioni trovate all'interno dell' Insieme Pareto Ottimale è quella più adatta a soddisfare le sue esigenze (il che può significare privilegiare un obiettivo piuttosto che un altro, o scegliere un valore intermedio in modo che tutti gli obiettivi siano ottimizzati allo stesso modo, e così via).

Un valido strumento per la risoluzione di problemi di ottimizzazione a più obiettivi sono gli Algoritmi Evolutivi, i quali si distinguono in Algoritmi Genetici, Strategie Evolutive ed Evolutionary Programming. Gli Algoritmi Evolutivi consistono in una classe di metodi di ottimizzazione che, simulando i processi dell'evoluzione naturale, e sfruttando le loro caratteristiche di robustezza ed abilità nel risolvere una vasta gamma di problematiche, sono in grado di evolvere

soluzioni per problemi del mondo reale.

I principi di base degli Algoritmi Genetici sono stati evidenziati per la prima volta da Holland nel 1975, nella sua opera Adaptation in Natural and Artificial Systems. Gli Algoritmi Genetici simulano quei processi naturali che sono essenziali per l'evoluzione naturale, quali la sopravvivenza del più forte, la ricombinazione genetica e la riproduzione.

Tali algoritmi sono quindi caratterizzati da tre operatori principali, che agiscono su di una popolazione iniziale di cromosomi: la Riproduzione selettiva degli individui migliori, la Ricombinazione Genetica (Crossover) e le Mutazioni casuali dei cromosomi. La bontà dei cromosomi viene valutata tramite una funzione, detta funzione di fitness. Quindi, soltanto gli individui con più alto valore di fitness saranno selezionati per la riproduzione, il crossover e la mutazione, ed avranno la possibilità di tramandare le proprie buone caratteristiche alle generazioni future.

Si ispirano agli Algoritmi Genetici le Strategie Evolutive, sebbene presentino delle differenze rispetto alla precedente classe di algoritmi. I principi di base sono uguali a quelli degli Algoritmi Genetici, tuttavia le Strategie Evolutive non fanno uso dell'operatore di crossover, ma sfruttano semplicemente la riproduzione selettiva e l'operatore di mutazione casuale. Esse si distinguono in due categorie: Strategie Evolutive a due membri e Strategie Evolutive multimembri.

Nelle strategie evolutive a due membri, ad ogni generazione, un singolo genitore genera un solo figlio, tramite il solo operatore di mutazione casuale. Invece, nelle Strategie Evolutive multimembri, più genitori sono utilizzati per produrre più figli, tramite gli operatori di selezione e mutazione.

Gli algoritmi Evolutionary Programming sono algoritmi evolutivi basati sull'operatore di mutazione che viene applicato ad uno spazio di ricerca discreto.

## 2.1 MultiObjective Optimization Problem

Parlare di Ottimizzazione per un dato problema significa trovare una o più soluzioni, cosiddette *fattibili*, che corrispondono ai valori estremi di una o più funzioni obiettivo. Quando un problema di ottimizzazione che modella un sistema fisico necessita di un'unica funzione obiettivo, il compito di trovare la soluzione ottimale è chiamato *Ottimizzazione ad un Singolo Obiettivo*. Al fine di estendere l'applicabilità di un algoritmo di ottimizzazione a differenti problemi,

sono stati imitati principi fisici e naturali così da sviluppare robusti algoritmi di ottimizzazione. Gli Algoritmi Evolutivi e il Simulated Annealing sono due esempi di tali algoritmi. Quando un problema di ottimizzazione richiede la presenza di più di una funzione obiettivo, il problema di trovare una o più soluzioni ottimali prende il nome di *Ottimizzazione Multiobiettivo*. Dal momento che l'Ottimizzazione Multiobiettivo necessita di più di un obiettivo, è immediato concludere che l'Ottimizzazione ad un Singolo Obiettivo è un caso particolare dell'Ottimizzazione Multiobiettivo. Comunque c'è una ragione per cui bisogna prestare più attenzione all'Ottimizzazione Multiobiettivo rispetto a quella ad un Singolo Obiettivo: la maggior parte dei problemi del mondo reale richiede la simultanea ottimizzazione di una serie di obiettivi che talvolta sono in competizione tra loro. Una soluzione ottimale rispetto ad un obiettivo richiede un compromesso con gli altri obiettivi.

Quando si ha un problema di ottimizzazione ad un singolo obiettivo (Single Objective Problem –SOP–), l'insieme delle soluzioni ammissibili è totalmente ordinato rispetto alla funzione obiettivo $\mathbf{f}$: per due soluzioni $\mathbf{a}$ e $\mathbf{b} \in \mathbf{X_f}$ si ha: $\mathbf{f(a)} \geq \mathbf{f(b)}$ oppure $\mathbf{f(b)} \geq \mathbf{f(a)}$.

L'obiettivo è trovare la soluzione (o le soluzioni) che danno il massimo valore di f. Invece, quando si hanno una serie di obiettivi, $\mathbf{X_f}$ in generale non è totalmente ordinato, ma parzialmente ordinato. Questo può essere spiegato attraverso la seguente definizione:

**Definizione 2.1.1** *Per due qualsiasi vettori obiettivo* $\mathbf{a}$ *e* $\mathbf{b}$ *possono verificarsi le seguenti condizioni:*

$$\begin{aligned} \mathbf{a} = \mathbf{b} \quad & \textit{se} \quad \forall \mathbf{i} \in \{1, 2, \ldots, k\} : a_i = b_i \\ \mathbf{a} \geq \mathbf{b} \quad & \textit{se} \quad \forall \mathbf{i} \in \{1, 2, \ldots, k\} : a_i \geq b_i \\ \mathbf{a} > \mathbf{b} \quad & \textit{se} \quad \mathbf{a} \geq \mathbf{b} \wedge \mathbf{a} \neq \mathbf{b}. \end{aligned} \tag{2.1}$$

*Le relazioni* $\leq$ *e* $<$ *sono definite analogamente.*

Quindi due vettori decisione $\mathbf{a}$ e $\mathbf{b}$ possono avere *tre* possibilità con i MOP rispetto alla relazione $\geq$ (al contrario di quel che accade per un SOP): $\mathbf{f(a)} \geq \mathbf{f(b)}$, $\mathbf{f(b)} \geq \mathbf{f(a)}$, oppure $\mathbf{f(a)} \ngeq \mathbf{f(b)}$ e $\mathbf{f(b)} \ngeq \mathbf{f(a)}$.

**Definizione 2.1.2** *Per due qualsiasi vettori decisione* $\mathbf{a}$ *e* $\mathbf{b}$*, si dice che:*

$$\begin{aligned} \mathbf{a} \succ \mathbf{b} \textit{ (a domina b )} \quad & \textit{se } \mathbf{f(a)} > \mathbf{f(b)} \\ \mathbf{a} \succeq \mathbf{b} \textit{ (a domina debolmente b )} \quad & \textit{se } \mathbf{f(a)} \geq \mathbf{f(b)} \\ \mathbf{a} \sim \mathbf{b} \textit{ (a è indifferente a b)} \quad & \textit{se } \mathbf{f(a)} \ngeq \mathbf{f(b)} \textit{ e } \mathbf{f(b)} \ngeq \mathbf{f(a)}. \end{aligned} \tag{2.2}$$

*Le definizioni per un problema di minimizzazione ($\prec$, $\preceq$, $\sim$) sono analoghe.*

Detto questo, è possibile introdurre il concetto di Pareto Ottimalità.

**Definizione 2.1.3** ***(Pareto Ottimalità)***
*Un vettore decisione* $\mathbf{x} \in \mathbf{X_f}$ *si dice non dominato rispetto all'insieme* $\mathbf{A} \subseteq \mathbf{X_f}$ *se:*

$$\nexists \mathbf{y} \in \mathbf{A} : \mathbf{y} \textit{ domina } \mathbf{x}. \tag{2.3}$$

*Inoltre,* $\mathbf{x}$ *si dice Pareto Ottimale se* $\mathbf{x}$ *è non dominato rispetto ad* $\mathbf{X_f}$.

L'insieme di tutte le soluzioni Pareto Ottimali prende il nome di *Insieme Pareto Ottimale*; i corrispondenti vettori obiettivo formano il *Fronte Pareto Ottimale*. Il generico vettore obiettivo $\mathbf{y} = \mathbf{f}(\mathbf{x})$, dove **x** è Pareto Ottimale, viene detto *efficiente*.

**Definizione 2.1.4** ***(Fronti e Insiemi non dominati)***
*Sia* $\mathbf{A} \subseteq \mathbf{X_f}$. *La funzione* $p(\mathbf{A})$ *dà l'insieme dei vettori decisione non dominati in* $\mathbf{A}$*:*

$$p(\mathbf{A}) = \{\mathbf{a} \in \mathbf{A} |\ \mathbf{a} \textit{ è non dominato rispetto ad } \mathbf{A}\}. \tag{2.4}$$

*L'insieme* $p(\mathbf{A})$ *è l'Insieme non dominato rispetto ad* $\mathbf{A}$*, il corrispondente insieme dei vettori obiettivo* $\mathbf{f}(p(\mathbf{A}))$ *è il Fronte non dominato rispetto ad* $\mathbf{A}$*. Inoltre, l'insieme* $\mathbf{X_p} = p(\mathbf{X_f})$ *è detto Insieme Pareto Ottimale e l'insieme* $\mathbf{Y_p} = \mathbf{f}(\mathbf{X_p})$ *è denotato Fronte Pareto Ottimale.*

L'Insieme Pareto Ottimale è l'insieme di tutte le soluzioni ottimali. Come accade per un problema ad un singolo obiettivo, anche nel caso dell'Ottimizzazione Multiobiettivo si può parlare di un *ottimo locale*, che costituisce l'insieme non dominato all'interno di un certo intorno. La corrispondenza tra Insieme Pareto Ottimale Locale e Globale è stata introdotta da Deb nel 1998-99:

**Definizione 2.1.5** *Si consideri un insieme di vettori decisione* $\mathbf{A} \subseteq \mathbf{X_f}$.

- *L'insieme* $\mathbf{A}$ *si dice Insieme Pareto-Ottimale Locale se*

$$\forall \mathbf{a} \in \mathbf{A} : \nexists \mathbf{x} \in \mathbf{X_f} : \mathbf{x} \succ \mathbf{a} \wedge \parallel \mathbf{x} - \mathbf{a} \parallel < \epsilon \wedge \parallel \mathbf{f}(\mathbf{x}) - \mathbf{f}(\mathbf{a}) \parallel < \delta \tag{2.5}$$

  *dove* $\parallel \cdot \parallel$ *è una metrica e* $\epsilon > 0$, $\delta > 0$.

- *L'insieme* $\mathbf{A}$ *si dice Insieme Pareto-Ottimale Globale se*

$$\forall \mathbf{a} \in \mathbf{A} : \nexists \mathbf{x} \in \mathbf{X_f} : \mathbf{x} \succ \mathbf{a}. \tag{2.6}$$

Si noti che un Insieme Pareto Ottimale Globale non necessariamente contiene tutte le soluzioni Pareto Ottimali e che ogni Insieme Pareto Ottimale Globale è a sua volta un Insieme Pareto Ottimale Locale.

Alla luce di quanto detto sopra, un problema di Ottimizzazione Multiobiettivo può essere definito nel seguente modo:

**Definizione 2.1.6** ***(Multiobjective Optimization Problem)***
*Un MOP include un insieme di n parametri (variabili decisione), un insieme di k funzioni obiettivo, ed un insieme di m vincoli. Le funzioni obiettivo ed i vincoli sono funzioni delle variabili decisione.*
*Gli obiettivi dell'ottimizzazione sono:*

$$\begin{aligned} &\textit{massimizzare} && \mathbf{y} = \mathbf{f}(\mathbf{x}) = (f_1(\mathbf{x}), f_2(\mathbf{x}), \ldots, f_k(\mathbf{x})) \\ &\textit{soggetto a} && \mathbf{e} = (e_1(\mathbf{x}), e_2(\mathbf{x}), \ldots, e_m(\mathbf{x})) \\ &\textit{dove} && \mathbf{x} = (x_1, x_2, \ldots, x_n) \in \mathbf{X} \textit{ e } \mathbf{y} = (y_1, y_2, \ldots, y_k) \in \mathbf{Y} \end{aligned} \tag{2.7}$$

*dove* $\mathbf{x}$ *è il vettore decisione,* $\mathbf{y}$ *è il vettore obiettivo,* $\mathbf{X}$ *è lo spazio decisione, ed* $\mathbf{Y}$ *è chiamato spazio obiettivo. I vincoli* $\mathbf{e}(\mathbf{x}) \leq \mathbf{0}$ *determinano l'insieme di soluzioni ammissibili.*

**Definizione 2.1.7** *L'insieme delle soluzioni ammissibili* $\mathbf{X_f}$ *è definito come l'insieme dei vettori decisione* $\mathbf{x}$ *che soddisfano i vincoli* $\mathbf{e}(\mathbf{x})$ *:*

$$\mathbf{X_f} = \{\mathbf{x} \in \mathbf{X} | \mathbf{e}(\mathbf{x}) \leq \mathbf{0}\} \tag{2.8}$$

*L'immagine di* $\mathbf{X_f}$*, ovvero la regione ammissibile nello spazio obiettivo, è denotata come:*

$$\mathbf{Y_f} = \mathbf{f}(\mathbf{X_f}) = \cup_{\mathbf{x} \in \mathbf{X_f}} \{\mathbf{f}(\mathbf{x})\} \tag{2.9}$$

Senza perdita di generalità, abbiamo considerato un problema di massimizzazione. Per un problema di minimizzazione le definizioni sono analoghe.

### 2.1.1 Ricerca e "Decision Making"

Nell'affrontare un problema di Ottimizzazione Multiobiettivo possono essere identificate due problematiche: *ricerca* e *decision making*.
Il primo di questi due aspetti si riferisce ad un processo di ottimizzazione nel quale l'insieme delle soluzioni fattibili è rappresentato dalle soluzioni Pareto Ottimali. Proprio come accade nel caso dell'ottimizzazione ad un singolo obiettivo, uno spazio di ricerca eccessivamente ampio può rendere complicato il raggiungimento di una soluzione ottimale del problema in questione. Il secondo aspetto (decision making) si riferisce al problema di scegliere la "migliore" soluzione all'interno dell'Insieme Pareto Ottimale. Colui che dovrà operare questa scelta prende il nome di *Decision Maker* (DM). Il DM è uno strumento necessario per bilanciare i vari obiettivi, che spesso sono in conflitto tra di loro.

I metodi di ottimizzazione multiobiettivo possono essere classificati in tre categorie, e questa suddivisione dipende da come sono combinate l'ottimizzazione e il processo di decisione:

**Decision Making prima della Ricerca** Gli obiettivi del MOP sono aggregati in un unico obiettivo che da solo racchiude le preferenze del DM.

**Ricerca prima della Decision Making** L'ottimizzazione viene eseguita senza che alcuna informazione di preferenza del DM venga data a priori. Il risultato della ricerca è un insieme di soluzioni (idealmente Pareto Ottimali), le quali sono candidate alla scelta finale che verrà operata dal DM.

**Decision Making durante la Ricerca** Il DM può manifestare le sue preferenze durante tutto il processo di ricerca, e quindi intervenire dopo ogni singolo passo dell'ottimizzazione.

## 2.2 Algoritmi Evolutivi

Il termine *Algoritmi Evolutivi* (EA) indica una classe di metodi di ottimizzazione che simulano i processi dell'evoluzione naturale [10]. Dopo un susseguirsi di molte generazioni, le popolazioni si evolvono secondo le leggi della selezione naturale e della sopravvivenza del più forte, come teorizzato per la prima volta da Darwin nella sua opera *L'origine della specie*. Imitando questi processi gli Algoritmi Evolutivi sono in grado di evolvere soluzioni per problemi del mondo reale.

I sistemi biologici assumono una notevole importanza grazie alla loro robustezza ed alla loro abilità nel risolvere una vasta gamma di problemi indispensabili alla loro sopravvivenza. Essi costituiscono il risultato di un processo evolutivo che pone le sue fondamenta su processi come la riproduzione selettiva degli individui migliori, la ricombinazione genetica dei loro cromosomi e su alcune mutazioni casuali. Nonostante l'esatto funzionamento dei principi dell'evoluzione naturale sia ancora oggetto di studio, i principi di base sono chiari:

- L'Evoluzione Naturale agisce sui cromosomi degli individui, piuttosto che sugli individui stessi, ovvero sulla codifica genetica (*genotipo*) delle caratteristiche fisiche dell'organismo vivente (che costituiscono invece il *fenotipo*).

- I processi di selezione naturale favoriscono la riproduzione degli individui (e quindi dei cromosomi) più efficienti dal punto di vista adattativo. In sostanza, in Natura gli individui di una popolazione competono tra loro per ricercare e ottenere risorse necessarie alla sopravvivenza, quali acqua, cibo e territorio. Inoltre spesso individui della stessa specie devono competere per la conquista di un compagno. Gli individui che risulteranno più adatti alla sopravvivenza e alla riproduzione avranno allora un maggior numero di discendenti. Quindi il processo di selezione è quello in cui il fenotipo influenza in qualche modo il genotipo, nonostante il fatto che le caratteristiche del fenotipo apprese in vita non restano impresse nel genotipo.

- Il meccanismo della riproduzione costituisce il nucleo del processo evolutivo: la combinazione dei codici genetici di due individui e l'introduzione di mutazioni casuali da un punto di vista adattativo. La combinazione delle buone caratteristiche di diversi antenati a volte può produrre una discendenza molto adattata (detta *superfit*), la cui qualità è di gran lunga superiore a quella di ciascun genitore. In questo modo le popolazioni si evolvono e diventano sempre più adattate al loro ambiente.

- L'evoluzione naturale opera su popolazioni di individui attraverso un processo generazionale che non possiede alcuna memoria storica, ma si basa esclusivamente sull'interazione tra ogni singolo individuo e l'ambiente ecologico in cui esso vive.

Come già detto, i processi biologici della selezione naturale possiedono caratteristiche di *robustezza*, *auto-organizzazione*, *efficienza*, *adattamento* e *complessità* perfettamente consone ai

compiti cui devono adempiere, caratteristiche queste che se incorporate nei sistemi artificiali costruiti dall'uomo potrebbero essere di grande utilità. Ma i metodi tradizionali di costruzione dei sistemi artificiali raramente riescono a dare risultati soddisfacenti a tal punto da poter essere paragonati alle prestazioni degli esseri viventi. Probabilmente uno dei motivi di questo fallimento è il fatto che i metodi tradizionali si basano su procedimenti di tipo analitico, come l'isolamento del problema, la definizione teorica, l'identificazione delle variabili e la derivazione di una soluzione specifica, che sono molto distanti dai principi dell'evoluzione naturale.

A differenza delle tecniche tradizionali, gli Algoritmi Evolutivi si basano su principi molto simili a quelli dell'evoluzione naturale, ed inoltre essi possiedono una doppia finalità: in primo luogo essi sono utili per comprendere più a fondo i processi di sviluppo filogenetico dei sistemi viventi, in quanto utilizzano un processo di simulazione, ed in secondo luogo vogliono condurre a tecniche artificiali che abbiano le stesse caratteristiche di robustezza ed adattabilità dei processi biologici, così da essere in grado di risolvere quei problemi che i metodi tradizionali non sono in grado di risolvere.

Poiché gli algoritmi evolutivi sono tecniche di ottimizzazione di funzioni, è lecito chiedersi quali siano i vantaggi che essi offrono rispetto ai metodi di ottimizzazione tradizionali. Nel 1989 Goldberg confrontò gli algoritmi genetici con i tre metodi tradizionali: il metodo di ottimizzazione basato sul *Calcolo Infinitesimale*, il *Metodo Enumerativo* e il metodo di *Ricerca Casuale*.

Il Metodo basato sul Calcolo Infinitesimale è il metodo classico, e si divide a sua volta in due classi: metodi di ricerca diretta e metodi di ricerca indiretta. Il *Metodo di Ricerca Diretta* è basato sulla creazione di un sistema di equazioni (di solito non lineari) e sulla soluzione analitica di ognuna di esse, ponendo il gradiente della funzione uguale a zero. Il *Metodo di Ricerca Indiretta* consiste invece nel partire da un punto qualunque dello spazio dei parametri della funzione, spostandosi quindi nella direzione del gradiente della funzione, o in altre direzioni opportunamente costruite. Spesso entrambi i metodi falliscono nella risoluzione di problemi tipici del mondo reale, in quanto i problemi reali raramente soddisfano la condizione matematica di differenziabilità della funzione, necessaria per il calcolo del gradiente. Inoltre essi operano partendo da un singolo punto dello spazio, ed alla fine al massimo riescono a trovare il massimo (minimo) della funzione più vicino, che è un massimo (minimo) locale. Il risultato della ricerca tramite questi metodi può essere mediocre quando la funzione da ottimizzare possiede

numerosi massimi (minimi) locali, come accade per la maggior parte dei problemi del mondo reale: infatti, una volta raggiunto un picco locale, è difficile spostarsi da questo per trovarne un altro.

I Metodi di Ricerca Enumerativi consistono nell'esame sistematico di uno spazio finito opportunamente discretizzato. Ogni punto dello spazio della funzione viene valutato, fino a quando non si trova il punto di massimo (minimo). Questi metodi sono molto più efficaci dei metodi basati sul calcolo infinitesimale, ma sono tuttavia inefficienti in quanto possono operare soltanto su spazi molto ridotti rispetto a quelli dei problemi reali.

I Metodi di Ricerca Casuale consistono nel valutare in successione diversi punti dello spazio, scelti in modo casuale, registrando di volta in volta i punti migliori. Essi sono molto simili ai metodi di ricerca enumerativa, tuttavia essi sono stati resi molto più efficienti grazie all'impiego delle tecniche di 'ricottura simulata' (*simulated annealing*). Anche gli Algoritmi Evolutivi fanno uso di strumenti di ricerca casuale, ma tutto il processo è guidato dalla riproduzione selettiva, ed inoltre è basato su una codifica dei parametri da ottimizzare piuttosto che sugli stessi parametri.

E' grazie a queste caratteristiche che gli Algoritmi Evolutivi godono di quelle doti di robustezza che consentono loro di risolvere problemi del mondo reale, che sono invece difficili da risolvere per altre tecniche. Gli Algoritmi Evolutivi possono essere applicati ad una vasta gamma di problemi, non sono soggetti ai requisiti di differenziabilità della funzione, non è loro necessaria l'esplorazione dell'intero spazio di ricerca e la memorizzazione dei punti migliori, ed inoltre operano in parallelo su una popolazione di soluzioni distribuite sulla superficie di ricerca. Inoltre essi sono dotati della *funzione di fitness*: essa funge da guida all'intero processo evolutivo.

Gli Algoritmi Evolutivi si distinguono in tre categorie: *Algoritmi Genetici* (GA), *Strategie Evolutive* (ES) ed *Evolutionary Programming* (EP). Di queste tre classi di algoritmi discuteremo in dettaglio nel seguente paragrafo la prima.

### 2.2.1 Algoritmi Genetici

I principi basilari degli Algoritmi Genetici sono stati evidenziati per la prima volta da Holland nel 1975.

Un algoritmo genetico è un metodo euristico di ricerca ed ottimizzazione, ispirato al principio della selezione naturale di Charles Darwin che regola l'evoluzione biologica.

Nel corso dell'esecuzione, l'algoritmo interviene a modificare ripetutamente una popolazione costituita da un certo numero di soluzioni (individui): a ciascuna iterazione, esso opera una selezione casuale di individui della popolazione corrente, impiegandoli per generare nuovi elementi della popolazione stessa, che andranno a sostituire un pari numero d'individui già presenti, e a costituire in tal modo una nuova popolazione per l'iterazione (o generazione) seguente. Tale successione di generazioni evolve verso una soluzione ottima del problema assegnato.

Gli algoritmi genetici sono applicabili alla risoluzione di un'ampia varietà di problemi d'ottimizzazione non indicati per gli algoritmi classici, compresi quelli in cui la funzione obiettivo è discontinua, non derivabile, stocastica, o fortemente non lineare.

Secondo i principi classici dell'evoluzione naturale, ogni individuo trasmette parte del suo patrimonio genetico ai propri discendenti; inoltre, sporadicamente nascono individui con caratteristiche non comprese tra quelle presenti nel corredo genetico della specie originaria, dando origine in tal modo a variazioni genetiche. Infine, gli individui con le qualità più adatte all'ambiente in cui si trovano hanno maggiori possibilità di sopravvivere e riprodursi.

Come detto all'inizio, la traduzione e l'estensione di tali principi, validi per i sistemi biologici, anche ai sistemi artificiali, si deve storicamente a John Holland. Dopo un non breve periodo di tempo, in cui il rilievo di tale lavoro non fu pienamente riconosciuto, l'impiego degli algoritmi genetici si è andato consolidando in ambito informatico, ingegneristico, finanziario ed ovviamente nel campo delle scienze sociali e naturali.

L'implementazione di un algoritmo genetico rispetta l'analogia esistente coi sistemi naturali, e prevede sempre alcune fasi fondamentali, che si elencano di seguito:

- una fase di selezione, in cui sono individuati gli individui da riprodurre, detti genitori, i quali contribuiscono alla generazione successiva della popolazione di soluzioni.

- una fase d'incrocio o riproduzione degli individui selezionati, in cui due genitori sono combinati in modo da formare opportunamente dei nuovi individui per la prossima generazione.

- infine, una fase di mutazione, nel corso della quale vengono apportati dei cambiamenti ca-

suali ai genitori prima che questi possano generare nuovi individui, che pertanto avranno un patrimonio genetico di caratteri diverso da quello dei genitori.

In dettaglio, l'algoritmo evolve attraverso i seguenti punti:

- l'algoritmo comincia generando, in maniera casuale, una popolazione iniziale;
- l'algoritmo crea in seguito una sequenza di nuove popolazioni, o generazioni. In ciascuna iterazione, gli individui della popolazione corrente sono usati per creare la generazione successiva, e a questo scopo si compiono degli ulteriori passi:
  - ciascun membro della popolazione corrente è valutato calcolandone il rispettivo valore di fitness (idoneità);
  - si determina un opportuno ordinamento di tali individui sulla base dei valori di fitness;
  - gli individui più promettenti sono selezionati come genitori;
  - a partire da tali individui si genera un pari numero di individui della generazione successiva, e ciò può avvenire secondo due modalità distinte, vale a dire effettuando cambiamenti casuali su un singolo genitore – mutazione – oppure combinando opportunamente le caratteristiche di una coppia di genitori – incrocio.
  - gli individui così generati vanno a sostituire i genitori consentendo la formazione della generazione successiva.
- infine, l'algoritmo s'interrompe quando uno dei criteri d'arresto è soddisfatto.

Un noto teorema, dovuto ancora a Holland, assicura che, sotto determinate ipotesi, gli individui con alti valori di fitness tendono a crescere esponenzialmente nella popolazione attraverso il meccanismo dell'incrocio, assicurando così la convergenza dell'algoritmo genetico verso una soluzione ottimale. Nel suo teorema sugli schemi (*schema theorem*), detto anche teorema fondamentale degli algoritmi genetici, egli dimostra che uno schema (ossia una particolare combinazione di geni che occupano posizioni precise all'interno di un cromosoma) prolifera più rapidamente se, oltre ad avere un alto valore di fitness, contiene un piccolo numero di geni specifici non lontani l'uno dall'altro. Ciò, infatti, riduce la probabilità di distruggere lo schema durante la fase di riproduzione.

Brevi successioni di geni, che assumono particolari valori, definiscono i cosiddetti blocchi costitutivi (*building blocks*): favorendo l'incrocio dei cromosomi meglio adattati, in cui si riscontra statisticamente la presenza di peculiari blocchi costitutivi, l'algoritmo aumenta la probabilità che blocchi costituenti opportuni, provenienti da cromosomi diversi, si ritrovino in uno stesso cromosoma. Assumendo che l'associazione di siffatti blocchi sia dunque vantaggiosa, dovrà anche ritenersi probabile la comparsa di un cromosoma (soluzione) eccellente per il problema in esame, in un tempo ragionevole.

La dimostrazione del teorema degli schemi è basata sull'ipotesi di codifica binaria, ma Wright (1991) l'ha estesa al caso di codifica con numeri reali; lo stesso Wright ha mostrato che una codifica reale è da preferirsi nel caso di problemi continui d'ottimizzazione. Herrera e Lozano (1998) hanno poi presentato un'ampia rassegna di operatori genetici applicabili a cromosomi codificati mediante numeri reali, compresi vari tipi di operatori di crossover (incrocio).

In base a un coefficiente stabilito inizialmente, alcune parti dei geni risultati migliori vengono scambiate, nell'ipotesi che questo possa migliorare il risultato della funzione di fitness nel successivo passo evolutivo. Ci sono varie tecniche di crossover. Una delle più semplici è la single point crossover (vedi figura 2.1) che consiste nel prendere due individui e tagliare le loro stringhe di codifica in un punto a caso. Si creano così due teste e due code. A questo punto si scambiano le teste e le code, ottenendo due nuovi geni. Il crossover non è applicato sempre, ma con una certa probabilità. Nel caso in cui non venga applicato i figli sono semplicemente le copie dei genitori.

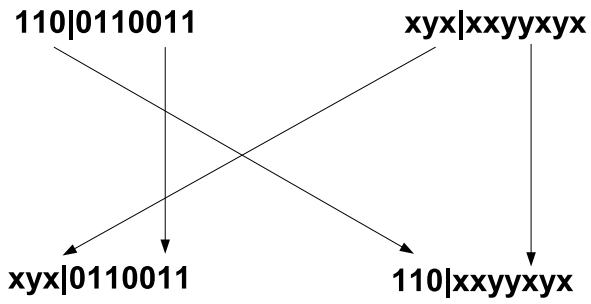



Figura 2.1: Single point crossover

Sperimentalmente si può vedere che il miglioramento diventa apprezzabile solo dopo un

certo numero di passi. Questo a meno di casi fortunati, ovviamente.

Il campo dei numeri reali costituisce ormai un'appropriata e consolidata forma di rappresentazione per gli algoritmi genetici in domini continui. Tuttavia, a causa di complessi fenomeni di interazione non lineare (epistaticità) tra gruppi di valori di una stringa rappresentante un individuo, non si può affermare con certezza che la combinazione di blocchi costitutivi altamente performanti sia sempre destinata a produrre individui ancora migliori. In altri termini, non sempre l'operazione genetica di crossover produce risultati accettabili, e anzi a volte accade che, a partire da due genitori estremamente promettenti, si ottenga un discendente decisamente meno valido.

# Capitolo 3

# Ottimizzazione OTA: MOEA

In questa applicazione test è stato effettuato il sizing di dispositivi MOS di una rete di circuito rappresentante un Operational Transconductance Amplifier a due stadi (two-stage OTA)mediante approccio multiobiettivo come alternativa all'approccio della funzione di costo, allo scopo di aumentare l'efficienza della progettazione dei dispositivi nei circuiti analogici.

Una formulazione multiobiettivo evita la necessità di pesare i differenti obiettivi in una singola funzione costo che non può ricoprire completamente l'analisi del problema.

Gli algoritmi utilizzati nella ottimizzazione sono stati gli algoritmi evolutivi multiobiettivo(Multi Objective Evolutionary Algorithm –MOEA– ), in particolare sono stati messi a confronto NSGA2 e SPEA2 con l'algoritmo MultiOb.

**NSGA II** è un algoritmo evolutivo con elitismo con una procedura di ordinamento non dominato e una stima di densità delle soluzioni effettuata da un meccanismo di distanza conosciuto come "crowding-distance".

**SPEA2** ha uno schema di assegnamento della funzione di fitness basato sulla relazione di Pareto Dominanza con una tecnica di stima della densità basata sul K-nn (K-th nearest neighbor).

L'algoritmo **MultiOb** è stato sviluppato dal Dipartimento di Ottimizzazione dell' Istituto ITWM ( – Institute Techno – und Wirtschaftsmatematik) Fraunhofer per la Matematica Industriale [14]. Questo algoritmo implementa una strategia evolutiva adattata a problemi specifici di ottimizzazione di performance di circuiti integrati (IC) secondo una metodologia multiobiettivo. Per gli scopi dell'ottimizzazione, è stata scelta una tecnica dalla famiglia di robuste meta–euristiche che supporta la generazione di soluzioni approssimate in relazione al criterio

di Pareto–dominanza.

## 3.1 Descrizione Algoritmi Evolutivi Multiobiettivo (MOEA)

Gli algoritmi evolutivi sono divenuti alcuni dei principali metodi per l'esplorazione del fronte pareto ottimale nei problemi di ottimizzazione multiobiettivo che risultano essere troppo complessi per essere risolti da metodi esatti, come quelli per la programmazione lineare o quelli basati sul calcolo del gradiente.

Ciò è dovuto non soltanto al fatto che ci sono poche alternative per la ricerca di soluzioni multiple pareto ottimali in grandi spazi "intrattabili".

Grazie al parallelismo, alla loro capacità di esplorare le similarità delle soluzioni con l'operazione della ricombinazione, tali algoritmi riescono ad approssimare il fronte di pareto ottimale in un singolo run di ottimizzazione. Questo fatto ha comportato un utilizzo degli algoritmi evolutivi multiobiettivo in sempre più numerose applicazioni e un interesse in rapida crescita verso questa tipologia di algoritmi.

### 3.1.1 NSGA II

L' NSGA II è un algoritmo genetico proposto da Kalyanmoy Deb nel 2000. La sigla NSGA II vuol dire *non-dominated sorting genetic algorithm II*, per sottolineare il fatto che in esso le soluzioni del problema di ottimizzazione in questione vengono ordinate in accordo al concetto di non dominanza. Tale algoritmo nasce dall'osservazione che tutti gli algoritmi evolutivi che usano il concetto di dominanza e la tecnica del *Fitness Sharing* presentano un alto grado di complessità computazionale, non fanno uso dell' elitismo ed hanno bisogno di specificare un parametro aggiuntivo, quale il $\sigma sh$. L'NSGA II è stato quindi realizzato per superare questi inconvenienti e, dagli studi svolti in passato, si è mostrato molto efficiente nella risoluzione di problemi di ottimizzazione multiobiettivo abbastanza complessi.

L'NSGA II può essere considerato il discendente di un altro algoritmo, l' NSGA (*non dominated sorting genetic algorithm*), proposto nel 1994 da Srinivas e Deb. L' NSGA è basato su severe regole di classificazione degli individui in quanto, prima della selezione, la popolazione viene classificata in base al concetto di dominanza: tutti gli individui non dominati vengono classificati in gruppi o classi, dette *rank*. Allo scopo di mantenere la diversità delle soluzioni,

gli individui classificati vengono suddivisi (shared) in base al valore che le funzioni di fitness assumono in essi e, una volta formato il primo gruppo, esso viene momentaneamente ignorato per procedere alla creazione di altri gruppi non dominati di ordine inferiore. Il processo continua fino a quando non vengono classificati tutti gli individui.

Questo algoritmo è stato criticato per tre motivi:

- Esso presenta un' alta complessità computazionale, la quale è $O(MN3)$, con $M$numero di obiettivi ed $N$grandezza della popolazione. Tale complessità dipende dalla complessità insita nel processo di ordinamento della popolazione.

- Questo rende l'NSGA dispendioso quando si usano popolazioni con un alto numero di individui.

- L'NSGA inoltre manca di elitismo e, usando il metodo di *fitness sharing* per assicurare la diversità della popolazione, richiede la specificazione del parametro $\sigma sh$.

**Non-domination sorting**

Molti algoritmi evolutivi che risolvono problemi di ottimizzazione multiobiettivo sono ideati per trovare solo il migliore fronte non dominato. Questi algoritmi classificano la popolazione in due insiemi: l'insieme non dominato ed il rimanente insieme dominato. Nell'NSGA II invece la popolazione ha bisogno di essere ordinata, come fatto nell'NSGA, in accordo ad un ordine ascendente di non dominanza, ovvero l'algoritmo opera una classificazione dell'intera popolazione, suddividendola in classi, o **ranks**. Le migliori soluzioni non dominate sono chiamate *non-dominated solutions* di rank 1, ovvero una volta trovati tutti gli elementi non dominati di una popolazione ad una determinata generazione, ad essi viene assegnata una rank pari ad 1. Tali elementi verranno poi momentaneamente trascurati per fare in modo che lo stesso procedimento venga applicato agli individui rimanenti, così da determinare la seconda rank, poi la terza, e così via. Alla fine del processo ogni individuo apparterrà ad una rank. La complessità di questa procedura è la somma delle complessità richieste nell'identificazione di ogni insieme non dominato.

E' importante notare che una volta identificato il primo insieme non dominato, il numero di soluzioni rimanenti è minore del numero originale di individui della popolazione, quindi le classificazioni successive alla prima richiederanno una complessità computazionale minore.

Nell'NSGA II il processo di classificazione ha complessità $O(MN2)$, con $M$ numero di obiettivi ed $N$ grandezza della popolazione.

In particolare nell' NSGA II per ogni soluzione vengono calcolate due entità:

*domination count*: il numero di soluzioni che dominano una generica soluzione p.

$Sp$: un insieme di soluzioni dominate dalla soluzione p.

Tutte le soluzioni del primo fronte non dominato avranno proprio il *domination count* uguale a 0. In seguito, per ogni soluzione p, con $\eta_p = 0$, verrà esaminato ogni membro $q$ del suo insieme $Sp$, il cui *domination count* verrà fissato uguale ad 1. Se nel fare ciò per ogni membro $q$ il *domination count* diventa 0, l'elemento $q$ verrà sistemato in una lista separata Q. Tali membri appartengono al secondo fronte non dominato. Questa procedura viene poi ripetuta con ogni membro di Q e il terzo fronte viene individuato. Un siffatto processo continua fino a quando non vengono identificati tutti i fronti o ranks.

Si nota quindi che per ogni soluzione p del secondo ordine di non-dominanza o di ordine più alto, il *domination count* $\eta_p$ può essere al più N-1; in questo modo ogni soluzione p sarà valutata N-1 volte prima che il suo domination count diventi 0. A questo punto alla soluzione considerata è assegnato un particolare livello di non-dominanza ed essa non sarà ulteriormente considerata.

Quindi, da quando rimangono al più N-1 soluzioni, la complessità di identificare il fronte di ordine due e i fronti di ordine superiore sarà $O(N2)$. Quindi la complessità totale della procedura sarà $O(MN2)$, con M il numero di obiettivi.

**Crowding-distance**

Una caratteristica fondamentale dell'NSGA II è quella di preservare la diversità tra le soluzioni dello stesso fronte non dominato, allo scopo di mantenere una buona distribuzione di soluzioni. Per fare ciò l'NSGA II si serve di una tecnica che lo distingue dagli altri algoritmi genetici: essa è la cosiddetta *crowding-distance*, per definire la quale dobbiamo prima definire una stima della densità delle soluzioni e un operatore, chiamato *crowded-comparison operator*.

Per dare una stima della densità di soluzioni che circondano una particolare soluzione, viene calcolata la distanza media di due punti, giacenti su ciascun lato del punto in questione, lungo ognuno degli obiettivi. Tale quantità serve come stima del perimetro di un cuboide formato usando come vertici i punti più vicini alla nostra soluzione. Essa sarà la crowding-distance.

Il calcolo dell Crowding-distance richiede che la popolazione venga ordinata, secondo un ordine crescente, in base al valore che ogni funzione obiettivo assume lungo le soluzioni. Quindi alle soluzioni dette *di confine*, ovvero quelle che presentano rispettivamente il più alto e il più basso valore delle funzioni, verrà assegnato un valore della crowding-distance infinito. Alle altre soluzioni viene invece assegnato un valore della crowding-distance uguale alla differenza dei valori della funzione di due soluzioni adiacenti. Il calcolo viene ripetuto per ogni funzione obiettivo e il valore complessivo della crowding-distance ,per ogni individuo, è calcolato come la somma dei valori di distanza corrispondenti ad ogni obiettivo.

A questo punto l'operatore di *crowded-comparison* guida il processo di selezione. Infatti fra due soluzioni con differenti ranks, si preferisce la soluzione con rank più bassa. Se le soluzioni appartengono alla stessa rank si preferisce quella che è collocata in una regione meno 'affollata', ovvero quella regione in cui il valore della crowding-distance è maggiore.

### Elitismo

Nel 1975 De Jong ha suggerito una strategia che consente di includere i migliori individui di una generazione nella successiva. Tale strategia, come già accennato, prende il nome di *elitismo* e fu provato dallo stesso De Jong che essa può migliorare il rendimento di un algoritmo genetico. L' NSGA II è un algoritmo elitista, cioè si è certi che i cromosomi migliori di una generazione saranno presenti anche nella generazione successiva. Perchè questo accada è necessario che tutti gli individui vengano ordinati, di generazione in generazione, secondo il concetto di pareto dominanza e per ognuno viene calcolata la suddetta *crowding-distance*. Quindi per assicurare l'elitismo, come sarà spiegato in dettaglio più avanti, ogni volta che si forma la *mating pool*, ovvero quella sorta di grande vasca in cui la popolazione dei genitori viene unificata con quella dei figli, vengono scelti ad ogni generazione gli individui con rank migliore, ovvero la più bassa, e con un alto valore di crowding-distance.

### Selezione

Il funzionamento di un algoritmo genetico, come già anticipato, si basa sull'uso degli operatori di selezione, di crossover e di mutazione. Per quel che riguarda la selezione l' NSGA II usa la cosiddetta **selezione per torneo**, durante la quale vengono disputate delle vere e proprie gare fra le soluzioni di una popolazione, scelte in maniera del tutto casuale. La soluzione migliore

viene poi selezionata per far parte della cosiddetta *mating pool*, ovvero una sorta di grande vasca all'interno della quale ogni soluzione è valutata per poi essere incrociata con altre soluzioni ed eventualmente mutata. Queste gare avvengono con due soluzioni per volta: la migliore avrà maggiore probabilità di riprodursi, mentre la peggiore verrà eliminata dalla popolazione.

Nell'NSGA II, mediante delle funzioni che determinano dei numeri casuali all'interno dell'intervallo [0,1], vengono generati due numeri che, opportunamente arrotondati, mediante la funzione *floor* - funzione predefinita del linguaggio C -, corrispondono a due differenti individui della popolazione, identificati appunto con un valore numerico. Di tali individui si mette a confronto la rank e la crowding-distance e agli individui con *rank* più bassa e con *crowding-distance* più alta verrà data maggiore probabilità di riprodursi, ovvero saranno selezionati per dare vita ai discendenti. Tale processo viene ripetuto tante volte quanti sono i membri della popolazione iniziale, cosicchè ogni individuo avrà la possibilità di partecipare ad almeno un torneo.

### Crossover

La creazione di nuovi individui è messa in atto dal **crossover**, ad opera del quale avviene uno scambio di materiale genetico. In particolare, se si usano variabili a codifica binaria l'NSGA II usa, in base alla scelta dell'utente, il crossover a punto singolo o il crossover uniforme. Nel *crossover a punto singolo* viene scelto, in maniera random, il cosiddetto punto di taglio a partire dal quale avviene lo scambio dei geni che costituiscono il cromosoma. Più in dettaglio nel crossover a punto singolo viene inizialmente generato un numero casuale compreso tra 0 ed 1. Se tale numero è minore della probabilità di crossover $p_c$ fissata dal decision maker, allora si procede con il crossover. Si genera quindi un altro numero random che,opportunamente arrotondato con la funzione predefinita *floor*, individua il cosiddetto **mating site**, ovvero il punto di accoppiamento. Se tale punto è al di fuori della lunghezza di ogni cromosoma, il punto di taglio viene individuato da un valore pari alla metà del numero precedentemente generato. Quindi fra ogni coppia di genitori avverrà uno scambio di materiale genetico, e verranno generati due figli che avranno la prima parte del proprio patrimonio genetico (che va dall'inizio del cromosoma al punto di taglio) uguale a quello di un genitore e la seconda parte (che va dal punto di taglio alla fine del cromosoma) uguale a quello dell'altro genitore.

Se invece il primo numero random generato risulta maggiore di $p_c$, il crossover non verrà

effettuato e li individui saranno semplicemente ricopiati. Questo procedimento viene effettuato un numero di volte pari alla metà degli individui della popolazione e considerando ogni volta un individuo e il successivo.

Nel *crossover uniforme* invece ogni figlio è generato scegliendo ogni gene con una certa probabilità da ogni genitore. Anche in questo caso il promo passo da fare è quello di generare un numero random compreso tra 0 e 1. Se tale numero è minore della probabilità di crossover, allora ogni gene del primo e del secondo figlio sarà sostituito dai geni del secondo genitore. Se il numero random invece è maggiore della probabilità di crossover, i due genitori verranno ricopiati.

Anche in questo caso, come nel crossover a punto singolo, il procedimento descritto viene effettuato un numero di volte pari alla metà degli individui della popolazione e considerando ogni volta un individuo e il successivo. Il crossover uniforme, a differenza di quello a punto singolo, potrebbe non preservare la struttura del genitore, producendo quindi un effetto alquanto distruttivo.

Nel caso in cui si usino variabili reali si usa invece il cosiddetto **SBX operator**, ovvero *crossover binario simulato*, che, come il nome stesso suggerisce, simula il funzionamento del crossover a punto singolo sulle stringhe binarie.

L'SBX lavora considerando due genitori alla volta e creando due figli. Di ogni genitore si mettono a confronto, in maniera sequenziale, gli elementi che costituiscono il cromosoma, in questo caso formato da numeri reali.

Si noti ancora che solitamente si vogliono creare dei figli più vicini ai genitori.

### Mutazione

L'operatore di **mutazione**, con una probabilità assegnata di solito molto bassa può alterare uno dei geni di un cromosoma. Nel caso in cui si usino variabili a codifica binaria, in cui i cromosomi sono stringhe di 0 ed 1, viene generato un numero random nell'intervallo [0,1]. Se esso è minore della probabilità di mutazione $p_m$ fissata dall'utente, ogni elemento della stringa sarà mutato, ovvero ogni 0 diventerà 1 ed ogni 1 diventerà 0. Questo procedimento viene eseguito per ogni elemento della popolazione. Se si lavora invece con variabili reali l'NSGA II usa la cosiddetta **mutazione polinomiale**.

L'NSGA II segue, nei caratteri generali, il seguente pseudo-codice:

**Algorithm 1** Algoritmo genetico NSGA II($d$, *gen*, $pc$, $pm$, *idcross*, *idmut* )

1: Inizializzare la popolazione
2: Generare la popolazione in modalita' random - grandezza $d$
3: Valutare le Funzioni Obiettivo
4: Assegnare la rank in base al concetto di Pareto-dominance–"sort"
5: **for** i=1 **to** gen **do**
6: Generare una popolazione di figli
7: Applicare l'operatore di selezione per torneo
8: Applicare gli operatori di ricombinazione e mutazione
9: Valutare le Funzioni Obiettivo per la nuova popolazione
10: Assegnare la rank in base al concetto di Pareto-dominance–"sort"
11: Con la popolazione di genitori e figli
12: Generare gruppi di fronti non dominati
13: Loop (interno) per aggiungere soluzioni alla nuova generazione cominciando dal primo fronte fino ai $d$ individui trovati
14: Determinare la crowding-distance fra i punti in ogni fronte
15: Selezionare i punti del fronte piu' basso, (ovvero con rank piu' bassa) e con un'alta crowding-distance (elitismo)
16: Incrementare l'indice relativo alla generazione
17: **end for**

Inizialmente viene creata una popolazione di grandezza $d$ (fissata dall'utente) in modalità random, distinguendo il caso in cui si usano variabili reali e quello in cui si usano variabili a codifica binaria.

Nel primo caso vengono generati cromosomi i cui elementi sono numeri reali, nel secondo caso vengono generati cromosomi i cui elementi sono 0 ed 1 e il cui valore reale sarà ottenuto mediante un processo di decodifica. In particolare nell'NSGA II si usa intanto, per ogni cromosoma, il classico metodo con cui si converte un numero in base 2 in un numero in base 10, interessando ogni gene che costituisce il cromosoma. Ovvero ogni gene verrà moltiplicato per una potenza di 2 il cui esponente è uguale alla posizione occupata dal gene nel cromosoma. Il numero risultante, per ogni gene, viene moltiplicato per la differenza fra l'estremo superiore e quello inferiore dell'intervallo di definizione della variabile corrente, a cui viene ancora som-

mato lo stesso estremo inferiore. Alla fine di tale processo si avrà quindi un nuovo cromosoma i cui geni sono numeri reali. Quindi si procede alla valutazione delle funzioni obiettivo, che coincidono con le funzioni che costituiscono il problema esaminato e, in accordo al concetto di Pareto dominanza, ad ogni individuo viene assegnata una classe, o rank, uguale al suo livello di 'non dominato' (1 è il livello migliore, 2 il livello seguente e così via). In particolare, come precedentemente detto, per identificare le soluzioni del primo livello non dominato ogni soluzione verrà confrontata con ogni altra soluzione della popolazione. Trovati tutti i membri del primo livello non dominato, essi saranno momentaneamente trascurati e il procedimento si ripeterà per trovare il secondo livello, e così tutti gli altri. Seguirà l'ordinamento della popolazione in ordine ascendente. A questo punto viene ripetuto un ciclo tante volte quante sono le generazioni fissate; durante tale ciclo verrà generata la popolazione dei discendenti, mediante i tre operatori di selezione per torneo, ricombinazione e mutazione descritti in precedenza. Quindi si valutano le funzioni obiettivo per ogni individuo della nuova popolazione e ad ognuno di essi sarà assegnata un rank, in accordo, ancora una volta, al concetto di Pareto-dominanza e la nuova popolazione verrà ordinata in ordine crescente in base al livello di non dominanza. Successivamente verrà considerata una popolazione formata da genitori e dai figli precedentemente generati; verrà creata cioè la cosiddetta *mating pool*. In essa la popolazione sarà ordinata in base alla rank di ciascun individuo e si formeranno così delle sottofamiglie, comunemente chiamate **fronti non dominati**, in ognuno dei quali si troveranno tutti gli individui appartenenti alla stessa rank. Quindi, non appena tutti i membri della precedente e della attuale popolazione vengono unificati, l'elitismo è assicurato e le soluzioni appartenenti al migliore fronte non dominato, ovvero al primo, saranno le migliori. Ora, se il numero degli individui appartenenti a questo fronte è minore di d (numero di individui necessariamente presente ad ogni generazione), per la successiva popolazione verranno intanto scelti tutti gli individui del primo fronte e i rimanenti membri della popolazione saranno scelti dai fronti successivi, scegliendo prima gli individui del secondo fronte, seguiti da quello del terzo e così via. E' importante notare che, allo scopo di selezionare esattamente d membri della popolazione, per ogni individuo sarà calcolata la crowding-distance e i membri dell'ultimo fronte saranno ordinati in ordine crescente mediante l'operatore di *crowded-comparison*. Quindi, per scegliere gli ultimi membri della popolazione verranno selezionati gli individui dell'ultimo fronte che presentano la rank più bassa ed un alto valore della crowding-distance. Intanto un indice relativo al numero di generazioni sarà incre-

mentato e il processo appena descritto sarà ripetuto fin quando l'indice non raggiungerà il valore fissato dall'utente, ovvero il numero di generazioni scelto.

Se si considera la complessità di ogni iterazione dell'intero algoritmo, ci si accorge che le complessità delle operazioni fondamentali sono le seguenti:

- la complessità dell'operazione di non-dominated sorting è:

$$O(M(2N)2);$$

- la complessità dell'assegnamento della crowding-distance è:

$$O(M(2N))\log(2N));$$

- la complessità dell'operazione di ordinamento tramite il crowded comparison operator è:

$$O(2N\log(2N)).$$

Quindi la complessità totale dell'algoritmo è $O(M(N)2)$, cioè risulta governata dall'operazione di non-dominated sorting.

Si noti ancora che la diversità delle soluzioni è introdotta con l'uso dell'operatore di *crowded comparison*, che viene applicato nella selezione per torneo e nella fase di riduzione della popolazione. Inoltre, poiché viene usata la crowding distance, non è richiesto nessun altro parametro di nicchia ($\sigma$*share*), il che rende l'NSGA II differente dagli altri algoritmi genetici.

### 3.1.2 SPEA2

Dopo i primi studi sull'ottimizzazione multiobiettivo attraverso gli Algoritmi Evolutivi, tra il 1993 e il 1994 sono state proposte alcune tecniche di ottimizzazione basate sulle tecniche di Pareto-dominanza, volte a dimostrare l'effettiva efficienza che tali algoritmi dimostrano nell'approssimare l'insieme di soluzioni ottimali anche in un singolo *run*. Come già anticipato, l'*elitismo* non è una caratteristica che questi algoritmi possiedono a priori, tuttavia è stato dimostrato sperimentalmente che algoritmi di ottimizzazione che si servono di questa tecnica sono più efficienti di altri. L'algoritmo, indicato sinteticamente con la sigla **SPEA2** (*Strength*

*Pareto Evolutionary Algorithm 2* ) incorpora nella sua struttura l'elitismo, come il suo immediato predecessore **SPEA**, presentato Da E. Zitzler e L. Thiele nel 1999, tuttavia è più efficiente di SPEA ed in grado di superare i limiti che tale algoritmo incontrava.

Quindi, innanzitutto vengono esposte brevemente le caratteristiche di SPEA, e successivamente si passerà alla descrizione peculiare di SPEA2.

**SPEA**

Poiché SPEA costituisce le fondamenta di SPEA2, è necessario illustrare, anche se in modo sommario, le caratteristiche di tale algoritmo. Per una descrizione più dettagliata è possibile consultare [26]. SPEA si serve di una regolare popolazione di cromosomi (*Pareto set*) e di un archivio (*external set*) ad ogni iterazione.

Si comincia con una popolazione iniziale ed un archivio vuoto, quindi si procede alle successive iterazioni. Gli elementi della popolazione iniziale vengono messi a confronto secondo il concetto di non-dominanza, quindi degli elementi non dominati si fa una copia all'interno dell'archivio, mentre tutti gli individui dominati o ripetuti sono rimossi dall'archivio durante questa fase di aggiornamento.

Se l'ampiezza dell'archivio è più grande di un certo limite (fissato a priori dal *decision maker*), l'archivio viene ridotto mediante una tecnica di *clustering* che mantiene le caratteristiche del fronte non-dominato. In generale, si parla di *clustering* quando, data una collezione di $p$ elementi, questi vengono suddivisi in $q$ gruppi di elementi relativamente omogenei (con caratteristiche pressoché identiche), dove $q < p$. A seconda dello schema di lavoro dell'algoritmo, è possibile distinguere due forme di *clustering*: il clustering diretto e il clustering gerarchico [Morse, 1980].

**Clustering diretto** ordina $p$ elementi in $q$ gruppi ad ogni *step*.

**Clustering gerarchico** lavora in modo interattivo collegando cluster adiacenti finché non si raggiunge il numero di gruppi richiesto.

SPEA utilizza il *clustering gerarchico*. Inizialmente, ognuno dei membri della popolazione forma un *cluster* ; al passo successivo, vengono selezionati due *cluster* ed incorporati il un *cluster* più grande, e questo processo continua finché non viene raggiunto il numero di *clusters* richiesto. Ovviamente di volta in volta i *cluster* da unire non vengono scelti a caso, ma secondo

un determinato criterio: vengono uniti quei *clusters* che si trovano tra loro più vicini. Una volta conclusa la suddivisione in gruppi (ovvero, in *clusters*), si seleziona un rappresentante da ciascuno di essi, in modo da ridurre la popolazione. Come soluzione rappresentativa si sceglie il *centroide* del *cluster*, ovvero il punto che ha la più piccola distanza media dagli altri punti dello stesso gruppo.

Una volta conclusa questa fase, ad ogni elemento della popolazione e ad ogni elemento dell'archivio si assegna un valore di *fitness*, come segue:

- Ad ogni individuo $i$ dell'archivio si assegna un valore di *strength* $S(i) \in [0, 1)$, che allo stesso tempo rappresenta il valore di fitness $F(i)$. $S(i)$ rappresenta il numero di membri della popolazione $j$ che sono dominati da $i$ o indifferenti ad $i$ riguardo alla relazione di dominanza, diviso la grandezza della popolazione più uno.

- Il valore di fitness $F(j)$ di un individuo $j$ della popolazione è calcolato sommando il valore di *strength* $S(i)$ di tutti i membri $i$ dell'archivio che dominano $j$ o sono indifferenti a $j$, aggiungendo uno alla fine.

Quindi segue la **selezione per accoppiamento**.

Nei più moderni EA per l'ottimizzazione multiobiettivo sono state sviluppate due tecniche di selezione: la **selezione ambientale**(Environmental selection) e la **selezione per accoppiamento**(mating selection):

**Selezione ambientale:** oltre alla popolazione viene mantenuto un archivio che contiene una rappresentazione del fronte non-dominato tra tutte le soluzioni considerate fino a quel momento. Un elemento dell'archivio viene rimosso soltanto se:

- è stata trovata una soluzione che lo domina o

- se la grandezza dell'archivio è stata superata e la porzione del fronte sulla quale sono collocati i membri dell'archivio è superaffollata.

**Selezione per accoppiamento:** si crea una grande 'vasca' di individui, che ad ogni generazione vengono valutati in due stadi. Dapprima tutti gli individui sono confrontati sulla base della relazione di Pareto dominanza, che definisce un ordine parziale su questo insieme. L'informazione che definisce se un individuo domina, è dominato o è indifferente rispetto agli altri

individui dell'insieme viene utilizzata per definire un rango (*rank* ) alla *vasca generazionale*. Successivamente questa rank viene ridefinita, incorporando informazioni relative alla densità degli individui.

In SPEA, come già anticipato, il tipo di selezione è la *selezione per accoppiamento*: tutti gli individui della popolazione e dell'archivio concorrono per essere selezionati per l'accoppiamento attraverso un vero e proprio *torneo*. Si noti che, in questa fase, ogni membro dell'archivio ha una probabilità di essere selezionato molto più alta di qualunque altro membro della popolazione. Infine, dopo il crossover e la mutazione, la vecchia popolazione viene rimpiazzata dalla popolazione della prole.

Nonostante SPEA si sia dimostrato abbastanza performante in diversi studi comparativi, esso presenta comunque delle carenze, che possono essere riassunte nei seguenti punti.

1. *Assegnazione del valore di fitness*: quegli individui che sono dominati dagli stessi membri dell'archivio hanno un identico valore di fitness. Questo significa che se l'archivio contiene un solo individuo, tutti i membri della popolazione avranno lo stesso valore di fitness, ovvero lo stesso *rank*, indipendentemente dal fatto che ci siano tra essi diverse relazioni di dominanza. Tale fatto compromette il buon esito del processo di ottimizzazione, perchè mancano informazioni sufficienti per la selezione, e SPEA in questo caso particolare funziona come un algoritmo di ricerca casuale.

2. *Stima della densità*: se molti individui della generazione corrente sono indifferenti, ovvero se nessuno di essi domina qualcun altro, non è possibile ricevere molte informazioni dalla relazione di dominanza, che, come già accennato, dovrebbe fornire un ordinamento parziale degli individui dell'insieme in questione. In tal caso, le informazioni relative alla densità saranno utilizzate per guidare il processo di ricerca, e poiché queste informazioni vengono utilizzate soltanto durante il *clustering* (che riguarda soltanto i membri dell'archivio), la popolazione viene totalmente ignorata.

3. *Troncamento dell'archivio*: sebbene la tecnica di *clustering* sia molto efficiente per ridurre l'insieme non dominato (l'archivio) senza distruggere le sue caratteristiche, questa può fare in modo che vengano perse delle soluzioni esterne.

### SPEA2: l'algoritmo

Proprio per superare le carenze di SPEA, è stato proposto un nuovo algoritmo genetico, SPEA2. In 2 è illustrato lo pseudo-codice di SPEA2.

Differentemente da SPEA, SPEA2 utilizza un metodo di assegnazione del valore di fitness detto di *fine grained*, che incorpora al suo interno le informazioni relative alla densità. Inoltre, ogni volta che il numero di individui non-dominati è minore della grandezza dell'archivio, questo viene riempito con gli individui dominati. Quindi mentre in SPEA la grandezza dell'archivio varia ad ogni iterazione, in SPEA2 la grandezza dell'archivio è un valore predefinito.

Oltretutto, la tecnica di *clustering*, che viene invocata ogniqualvolta il numero di individui non-dominati è più grande della grandezza dell'archivio, è stata rimpiazzata con una alternativa tecnica di troncamento, che è simile alla precedente ma non permette che vengano perse le soluzioni più esterne. Infine, a differenza di quel che accade in SPEA,in SPEA2 partecipano al processo di selezione per accoppiamento soltanto i membri dell'archivio.

**Algorithm 2** Algoritmo SPEA2 ( $N$(grandezza della popolazione); $T$(numero massimo di generazioni))

1: **Step 1: Inizializzazione:** Genera una popolazione iniziale $P0$ e crea un archivio vuoto (*external set*) $\underline{P0} = \emptyset$. Poni $t$= 0.

2: **Step 2: Assegnazione del valore di fitness:** Calcola il valore di fitness degli individui in $Pt$e in $\underline{Pt}$.

3: **Step 3: Selezione ambientale:** Copia tutti gli individui non-dominati di $Pt$ e $\underline{Pt}$ in $\underline{Pt+1}$. Se $|\underline{Pt+1}| > |\underline{N}|$, allora riduci $Pt+1$ mediante l'operatore di troncamento, altrimenti, se $|\underline{Pt+1}| < |\underline{N}|$, riempi $Pt+1$ con gli individui dominati di $Pt$ e di $\underline{Pt}$ .

4: **Step 4**: **Criterio di Stop:** Se $t \geqslant T$, oppure e'soddisfatto un altro criterio di stop, allora metti l'insieme dei vettori decisione rappresentati dagli individui non-dominati $A$ in $\underline{Pt+1}$. Stop.

5: **Step 5: Selezione per accoppiamento:** Esegui la selezione per torneo (*binary tournament*) rimpiazzando gli individui di $\underline{Pt+1}$ in modo da riempire la *mating pool*.

6: **Step 6: Variazione:** Applica gli operatori di ricombinazione genetica e mutazione agli individui nella *mating pool* e indica con $Pt+1$ la popolazione risultante. Poni $t = t$+ 1 e vai allo *Step 2*.

### 3.1.3 Software MultiOb

L'algoritmo descritto in questa sezione è stato sviluppato dal Dipartimento di Ottimizzazione dell' Istituto ITWM ( – Institute Techno – und Wirtschaftsmatematik) Fraunhofer per la Matematica Industriale [14].

Questo algoritmo implementa una strategia evolutiva adattata a problemi specifici di ottimizzazione di performance di circuiti integrati (IC) secondo una metodologia multiobiettivo.

In particolare, da un punto di vista tecnico il problema è considerato come un problema di ottimizzazione multiobiettivo continuo con vincoli che diano limiti superiori ed inferiori rispetto alle variabili di decisione (valori parametrici), formalmente:

$$\min_{\mathbf{x} \in H \subseteq \mathbb{R}^n} \mathbf{f}(\mathbf{x}) \tag{3.1}$$

con $f_i : \mathbb{R}^n \leftarrow \mathbb{R}$ funzioni continue. In questa formulazione, $n$ è il numero di parametri (variabili decisionali), $q$ è il numero delle funzioni obiettivo, $\mathbf{f}$ è il vettore dei valori delle funzioni obiettivo, $H$ è un iper-rettangolo nello spazio di definizione del parametri.

Si assume che le funzioni siano continue Lipschitziane. Si assume inoltre che il numero delle variabili di decisione, $n$, sia tra 10 e 50 e il numero di funzioni obiettivo, $q$, sia minore o uguale a 10. Non ci sono comunque restrizioni sul numero di variabili e funzioni obiettivo considerate nel software.

L'ottimizzazione viene eseguita utilizzando il concetto di black–box. Questo significa che l'algoritmo di ottimizzazione può non avere nessun vantaggio dal conoscere la formulazione esplicita delle funzioni obiettivo ma applica la simulazione numerica o simbolica del circuito (o i suoi risultati) come un mezzo esterno per valutare le funzioni obiettivo.

Per gli scopi dell'ottimizzazione, è stata scelta una tecnica dalla famiglia di robuste meta–euristiche che supporta la generazione di soluzioni approssimate in relazione al criterio di Pareto–dominanza.

**Definizione 3.1.1 (Pareto–dominanza)** *Dati* $\mathbf{y}', \mathbf{y}'' \in \mathbb{R}^n$, $\mathbf{y}'$ *domina* $\mathbf{y}''$ *se:*

$$(\forall 1 \leq i \leq m \ : \ y'_i \leq y''_i) \wedge (\exists j \ : \ y'_j < y''_j) \tag{3.2}$$

Questa tecnica di ottimizzazione multiobiettivo è basata su algoritmi evolutivi (EA) adattati a problemi di ottimizzazione specifici.

L'algoritmo utilizza un insieme di parametri per l'algoritmo evolutivo e per gli operatori utilizzati all'interno dell'algoritmo evolutivo. I parametri specifici dell'algoritmo evolutivo sono principalmente la dimensione della popolazione (il numero di genitori *parents* $\mu$ , ed il numero di figli *offspring population* $\lambda$), il numero massimo di generazioni da eseguire, la strategia di selezione da eseguire (con o senza elitismo) e il tipo di ottimizzazione (massimizzazione o minimizzazione). Mentre i parametri utilizzati specificatamente nelle funzioni che effettuano le operazioni proprie dell'algoritmo evolutivo sono la probabilità di mutazione, la probabilità di ricombinazione, un parametro che specifica il rate di mutazione, la dimensione del passo medio per gli aggiornamenti della probabilità di ricombinazione. In generale la definizione di questi parametri influenza la performance dell' EA e lo sforzo computazionale. In pratica, la qualità dell'approssimazione migliora con l'aumento della dimensione della popolazione e del numero di generazioni, mentre lo sforzo computazionale cresce linearmente con ciascuno dei parametri specifici relativi agli operatori dell'EA e in generale i valori ammessi per tali parametri vengono determinati sui problemi test effettuando diversi esperimenti.

L'algoritmo permette una scelta sul criterio di selezione per la generazione della nuova popolazione, tale criterio può essere di selezione mediante un unico obiettivo per problemi di tipo SOP (Single Objective Problem), generazione di soluzioni approssimate in relazione al criterio di Pareto–dominanza, generazione di soluzioni effettuate mediante il calcolo di una unica funzione obiettivo risultato dalla somma pesata delle singole funzioni obiettivo. Inoltre la selezione può prevedere o meno la inclusione dei genitori, la strategia che prevede l'inclusione dei genitori nella selezione permette una selezione più elitaria che non permette la deteriorazione una volta ottenuta una soluzione.

L'implementazione di tale algoritmo evolutivo permette quindi l'unificazione di approcci differenti per realizzare il passo di selezione multiobiettivo nell'algoritmo evolutivo. Ciò è possibile implementando differenti medodi di multicriteria decision making (MCDM) come classi object-oriented. Ciò significa che questi metodi sono disponibili come oggetti muniti di una interfaccia comune ben-definita per la loro applicazione. Ciascuno di tali metodi MCDM può quindi essere utilizzato per la valutazione della funzione di fitness all'interno del passo di selezione. La connessione tra l'algoritmo evolutivo multiobiettivo e un particolare metodo MCDM può essere gestito in modo molto flessibile.

Come risultato, le routines restituiscono insiemi di soluzioni equivalenti in accordo alla

relazione di Pareto dominanza, cioè approssimazioni dell'insieme efficiente. Le soluzioni vengono restituite come output numerico standard formattate in uno stream di I/O. Vengono inoltre restituite alcune statistiche sulla soluzione ottenuta come il valore medio, minimo e massimo del valori delle funzioni obiettivo.

Il linguaggio di programmazione utilizzato è il C++. L'implementazione comprende diversi header file e source file per le routines di ottimizzazione multiobiettivo. Si tratta per lo più di implementazione di classi per il trattamento di problemi (che corrispondono alle istanze del circuito), soluzioni (che corrispondono ai valori dei parametri computati ), popolazioni (insieme di soluzioni usate dall'algoritmo evolutivo), algoritmi evolutivi (istanze parametrizzate di tali algoritmi) e matrici (alcune routines di base per il trattamento delle matrici). Qui di seguito viene riportata una lista dei files con una breve descrizione:

- `copt.cpp` - Main file del progetto
- `basic.cpp` - File contenente alcune routine di base e la classe matrix
- `basic.h` - Header file relativo alle routines di base e la classe matrix
- `ea.cpp` - Codice per le classi relative all'algoritmo evolutivo e alla popolazione
- `ea.h` - Header file relativo all'algoritmo evolutivo e alla popolazione
- `solution.cpp` - Classi di soluzione,problema ed ottimizzazione del circuito
- `solution.h` - Header file per le classi di soluzione,problema ed ottimizzazione del circuito

Qui di seguito è mostrato lo pseudo-codice dell'algoritmo evolutivo MultiOb.

**Algorithm 3** Algoritmo evolutivo MultiOb($pop\_size$ ,$pop\_off$,$n\_gen$)

1: Inizializza la popolazione di $pop\_size$ individui in maniera random
2: Valuta le Funzioni Obiettivo su tale popolazione
3: **for** i=1 **to** $n\_gen$ **do**
4: Genera una popolazione di dimensione $pop\_off$ mediante l'operatore di mutazione
5: Crea una nuova popolazione mediante l'operatore di ricombinazione.
6: Valuta le Funzioni Obiettivo per la nuova popolazione
7: Effettua la selezione in base al criterio prescelto: Pareto–dominanza.
8: Verifica se deve essere aggiornato il parametro per il rate di mutazione.
9: **end for**
10: Salva i risultati includendo alcune statistiche sui dati ricavati.

### Testing sull'algoritmo: MOP

Uno studio preliminare dell'algoritmo è stato effettuato su alcune funzioni di test abbastanza semplici, già utilizzate da numerosi studiosi per avvalorare la robustezza e l'efficienza degli algoritmi genetici. Queste funzioni sono state chiamate MOP. Le MOP rappresentano problemi di ottimizzazione multiobiettivo che presentano caratteristiche differenti: esse possono essere continue o discontinue, convesse o non convesse, unimodali o multimodali, deterministiche o stocastiche. Si è testato il funzionamento di MultiOb solo su quelle MOP che rappresentano un problema di ottimizzazione a due obiettivi, in modo da poter graficare il risultato ed avere un idea del funzionamento dell'algoritmo. Nulla vieta, ovviamente, di estendere MultiOb a problemi di ottimizzazione con più di due obiettivi. Il Fronte di Pareto che si delinea in seguito all'ottimizzazione di queste funzioni giace nello spazio degli obiettivi e può essere connesso o disconnesso, concavo o convesso. Tutte le funzioni, rappresentate nella tabella 3.1, tranne la MOP3, rappresentano problemi di minimizzazione.

Tabella 3.1: MOEA Test Suite Function

| MOP | Definizioni | Vincoli |
|---|---|---|
| MOP1 | $F = (f_1(x), f_2(x))$, dove<br>$f_1(x) = x^2$<br>$f_2(x) = (x-2)^2$ | $-10^5 \leq x \leq 10^5$ |
| MOP2 | $F = (f_1(\vec{x}), f_2(\vec{x}))$, dove<br>$f_1(\vec{x}) = 1 - exp(-\sum_{i=1}^{n}(x_i - \frac{1}{\sqrt{n}})^2)$<br>$f_2(\vec{x}) = 1 - exp(-\sum_{i=1}^{n}(x_i + \frac{1}{\sqrt{n}})^2)$ | $-4 \leq x_i \leq 4; i = 1, 2, 3$ |
| MOP3 | Massimizza $F = (f_1(x,y), f_2(x,y))$, dove<br>$f_1(x,y) = -[1 + (A_1 - B_1)^2] - (A_2 - B_2)^2$<br>$f_2(x,y) = -[(x+3)^2 + (y+1)^2]$ | $-3.1416 \leq x, y \leq 3.1416$<br>$A_1 = 0.5\sin 1 -$<br>$2\cos 1 + \sin 2 -$<br>$1.5\cos 2$<br>$A_2 = 1.5\sin 1 -$<br>$\cos 1 + 2\sin 2 -$<br>$0.5\cos 2$<br>$B_1 = 0.5\sin x -$<br>$2\cos x + \sin y -$<br>$1.5\cos y$<br>$B_2 = 1.5\sin x -$<br>$\cos x + 2\sin y -$<br>$0.5\cos y$ |
| MOP4 | $F = (f_1(\vec{x}), f_2(\vec{x}))$, dove<br>$f_1(\vec{x}) = \sum_{i=1}^{n-1}(-10e^{(-0.2)*\sqrt{x_i^2 + x_{i+1}^2}})$<br>$f_2(\vec{x}) = \sum_{i=1}^{n}(\mid x_i \mid^a + 5\sin(x_i)^b)$ | $-5 \leq x_i \leq 5$<br>$a = 0.8$<br>$b = 3$ |
| MOP6 | $F = (f_1(x,y), f_2(x,y))$, dove<br>$f_1(x,y) = x$<br>$f_2(x,y) = (1 + 10y)*$<br>$[1 - (\frac{x}{1+10y})^\alpha - \frac{x}{1+10y} sin(2\pi q x)]$ | $0 \leq x, y \leq 1$<br>$q = 4$<br>$\alpha = 2$ |

**MOP1**

La MOP1 è la prima funzione di Shaffer. Si tratta di una funzione a due obiettivi, ed è stata selezionata in particolare per la sua importanza storica. Infatti, quasi tutti gli algoritmi evolutivi sono stati testati su tale funzione. Il suo Fronte di Pareto è una singola curva convessa. La figura 3.1 mostra il risultato dell'esecuzione dell'algoritmo evolutivo MultiOb sulla funzione Test MOP1.

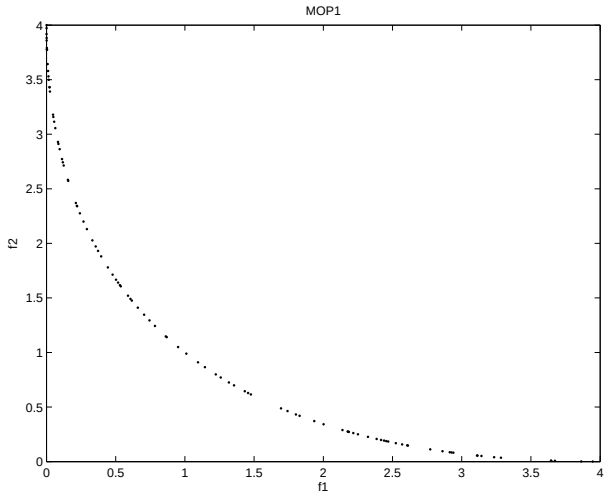



Figura 3.1: MOP1, $pop\_size$=100, $n\_generazioni$=1000, $p\_m$=0.01.

**MOP2**

La MOP2 è la seconda funzione di Fonseca. Si tratta di una funzione a due obiettivi, dove ciascun obiettivo dipende da tre variabili. Il suo Fronte di Pareto anche in questo caso è rappresentato da una curva continua, ma concava. La figura 3.2 mostra il risultato dell'esecuzione dell'algoritmo evolutivo MultiOb sulla funzione Test MOP2.

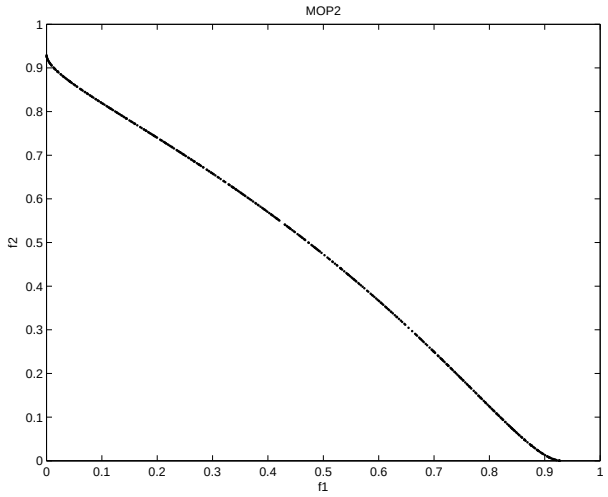



Figura 3.2: MOP2, $pop\_size$=500,$n\_generazioni$=10000, $p\_m$=0.04.

**MOP3**

La MOP3 è la funzione di Poloni, che, al contrario delle altre sei, rappresenta un problema di massimizzazione. Si tratta di una funzione a due obiettivi, ciascuno dipendente da due variabili. Inoltre sono presenti 4 vincoli. Il Fronte di Pareto è costituito da due curve tra le quali è

presente un salto di discontinuità. La figura 3.3 mostra il risultato dell'esecuzione dell'algoritmo evolutivo MultiOb sulla funzione Test MOP3.

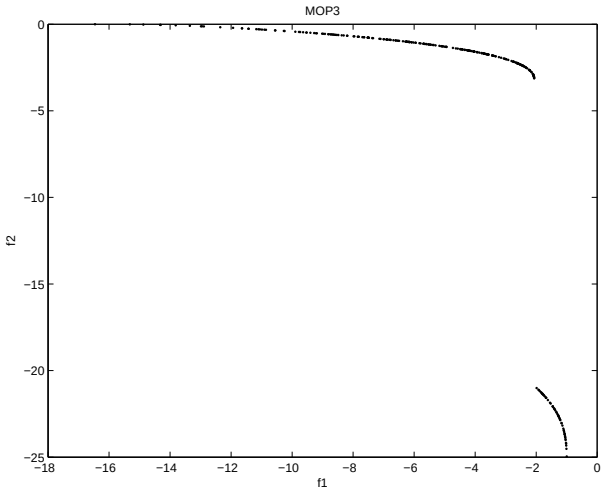



Figura 3.3: MOP3, $pop\_size$=300, $n\_generazioni$=1000, $p\_m$=0.03.

**MOP4**

La MOP4 è la funzione di Kurasawe. Si tratta di una funzione a due obiettivi e, come la MOP2, ciascun obiettivo dipende da tre variabili. Inoltre essa è sottoposta a due vincoli. Il suo Fronte di Pareto è una curva che presenta quattro salti di discontinuità. La figura 3.4 mostra il risultato dell'esecuzione dell'algoritmo evolutivo MultiOb sulla funzione Test MOP4.

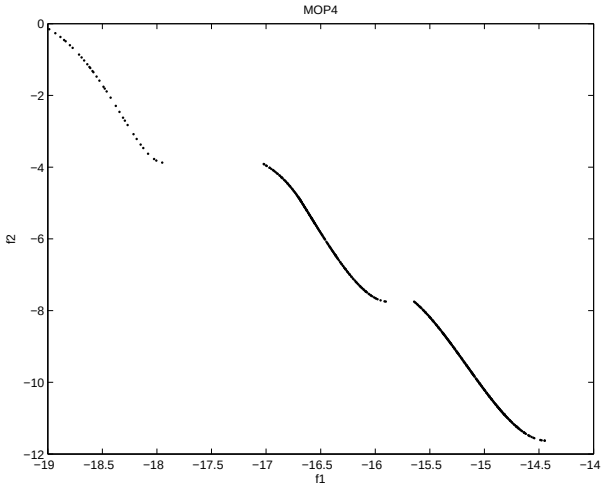



Figura 3.4: MOP4, $pop\_size$=500, $n\_generazioni$=3000, $p\_m$=0.04.

**MOP6**

Per quanto riguarda la MOP6, questa è una funzione a due obiettivi, ciascuno dipendente da

due variabili, ed è inoltre sottoposta a due vincoli. Il suo Fronte di Pareto è costituito da quattro curve. La figura 3.5 mostra il risultato dell'esecuzione dell'algoritmo evolutivo MultiOb sulla funzione Test MOP6.

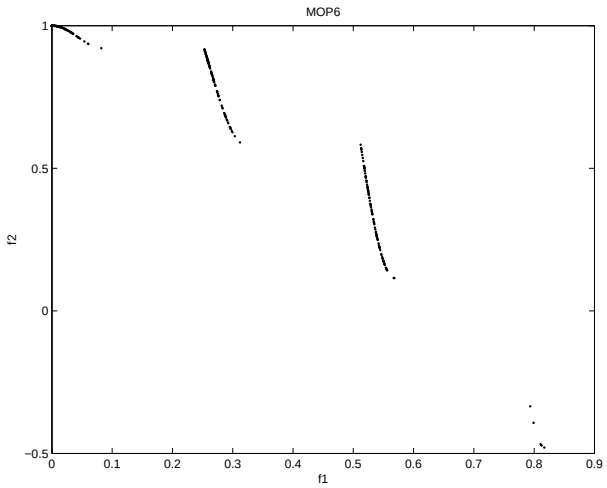



Figura 3.5: MOP6, $pop\_size$=300, $n\_generazioni$=1000, $p\_m$=0.01.

## 3.2 Descrizione del problema di ottimizzazione

Per quanto riguarda il processo di sizing di dispositivi MOS della rete di circuito rappresentante l'OTA a due stadi mediante Algoritmi Evolutivi Multiobiettivo, i parametri da ottimizzare e i loro range sono mostrati nella tabella 3.2. I parametri indicati con "W" si riferiscono alla larghezza di canale dei MOS, "L" si riferisce alla lunghezza di canale dei MOS, "R" (resistenza) e "C" (capacità).

Tabella 3.2: Parametri da ottimizzare per l'applicazione OTA.

| **Parametri** | **Ranges** | **Unità** |
|---|---|---|
| **W1b=W1a** | 7-20 | $\mu m$ |
| **W3** | 7-20 | $\mu m$ |
| **W5** | 7-20 | $\mu m$ |
| **L** | 0.525-0.875 | $\mu m$ |
| **C** | 3-5 | $pF$ |
| **R** | 20-40 | $K\Omega$ |
| **W4** | 7-20 | $\mu m$ |
| **W2b=W2a** | 7-20 | $\mu m$ |
| **I** | 1-15 | $\mu A$ |

Come visto nel Capitolo precedente, le metriche di performance nella progettazione di amplificatori analogici sono molteplici, ma, per le ottimizzazioni effettuate mediante i MOEA, sono state considerate le seguenti [9]:

- **Guadagno alle basse frequenze** : si tratta del guadagno a 100Hz che è la base per un range di amplificazione.
- **Margine di fase** : il margine di fase è una misura di qualità per il circuito perché è relativo agli effetti parassiti come l'accoppiamento incrociato che causa il fallimento nel raggiungimento delle performance richieste.
- **Frequenza all'unità di guadagno** : definita come il range di frequenza quando l'amplificatore ha almeno guadagno unitario.

- **Area** : L'area di circuito è una specifica importante per la progettazione perché è relativa allo Yield del processo di fabbricazione. Generalmente, se è possibile accrescere il numero di circuiti per unità di area allora lo Yield del processo di fabbricazione aumenta e il rapporto tra il numero di fallimenti funzionali e il numero totale dei circuiti per unità di area diminuisce. Nel caso dell'applicazione test, è stata data una stima inferiore data dalla somma delle larghezze dei MOS.

- **Consumo di potenza** : Il consumo di potenza è oggi molto importante per quanto riguarda il gran numero di applicazioni che funzionano a batteria.

Tale problema di ottimizzazione è stato quindi formulato come problema multiobiettivo in cui le funzioni obiettivo da ottimizzare sono appunto le metriche di performance che vanno di certo in conflitto tra loro.

Nella tabella 3.3 vengono mostrati dunque gli obiettivi da massimizzare/minimizzare:

Tabella 3.3: Obiettivi per il problema di ottimizzazione.

| **Obiettivi** | **Specifiche** |
|---|---|
| Consumo di potenza | Minimizzare |
| Larghezza totale | Minimizzare |
| Frequenza all'unità di guadagno | Massimizzare |
| Guadagno a 100 Hz | Massimizzare |
| Margine di fase | Massimizzare |

Inoltre, le specifiche minime per le performance sono state espresse in forma di vincoli, e vengono presentate nella tabella 3.4:

Tabella 3.4: Vincoli per le funzioni obiettivo

| **Obiettivi** | **Specifiche** | **Unità** |
|---|---|---|
| Frequenza all'unità di guadagno | $>$ 31.221 | $MHz$ |
| Guadagno a 100Hz | $>$ 64.118 | $dB$ |
| Margine di fase | $>$ 60 | $Degree$ |

## 3.3 Risultati

In questa sezione viene fatto un confronto tra NSGA2, SPEA2 e MultiOb per la ottimizzazione multiobiettivo dell'OTA.

L'ottimizzazione guida un flusso di simulazioni che è basato su una iterazione dell'algoritmo di simulazione con il simulatore circuitale Spice. Ogni iterazione è una valutazione del circuito che consiste di una o più simulazioni. Il criterio di stop utilizzato dagli algoritmi evolutivi e da MultiOb è il massimo numero di valutazioni della funzione $T_{max}$, anche chiamate valutazioni della funzione di fitness (ffe – fitness function evaluations).

I parametri di NSGA2 e SPEA2 sono:

**la probabilità di crossover** $p_c$=0.9;

**la probabilità di mutazione** $p_m$=1.

Entrambi gli algoritmi evolutivi utilizzano il crossover SBX con indice $\eta_c$=2 e mutazione gaussiana con deviazione standard $\sigma \in \{0.01, 0.1, 0.4\}$.

Questi valori dei parametri sono stati ottenuti da calcoli preliminari dei parametri, il parametro di mutazione gaussiano sembra essere il parametro di controllo più rilevante [3].

Ciascun algoritmo viene terminato dopo 600000 valutazioni della funzione.

I grafici rappresentati nelle figure 3.6 e 3.7 mostrano il trade-off tra triple di obiettivi.

Questi grafici caratterizzano i differenti compromessi realizzati nella ricerca delle soluzioni da parte degli algoritmi.

E' possibile notare gli effetti di crescita/decrescita di una performance in relazione alle altre e le dipendenze tra le differenti specifiche.

Per esempio si assume che il consumo di potenza cresce quando cresce la frequenza all'unità di guadagno e il margine di fase come si può vedere nelle proiezioni in due dimensioni delle figure 3.8.

In particolare, per quanto riguarda la funzione obiettivo "Frequenza" , NSGA2 e SPEA2 ricercano soluzioni non dominate nel range tra 30 e 75 MHz. Si notino in particolare le differenti distribuzioni di punti e in particolare la diffusione del fronte di Pareto. Questo fenomeno è relativo alla convessità o non convessità del problema multiobiettivo in regioni del dominio differenti.

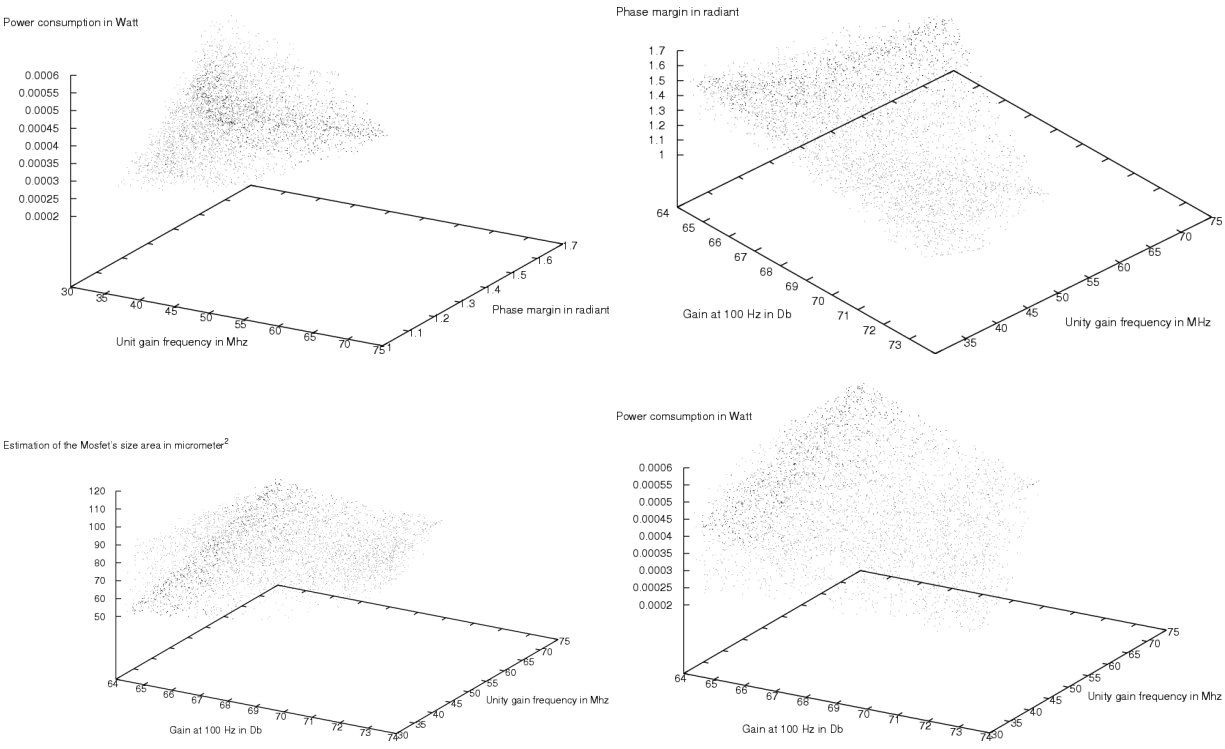



Figura 3.6: Trade-off di SPEA2.

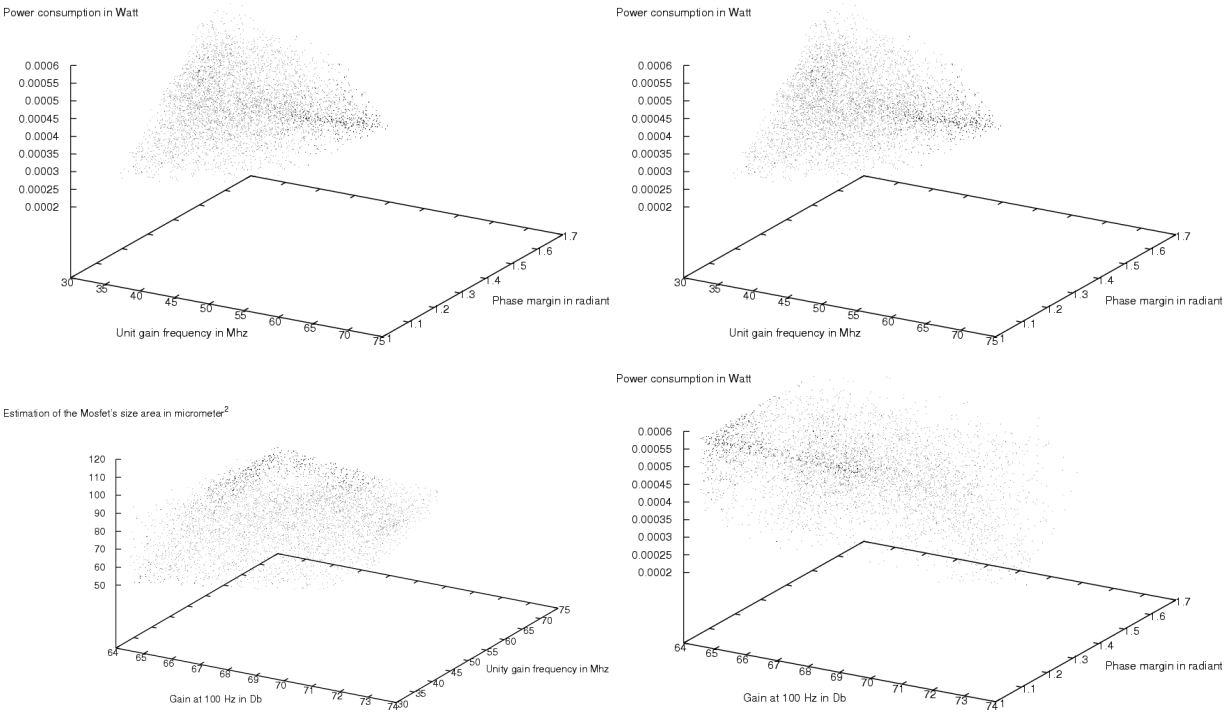



Figura 3.7: Trade-off di NSGA2.

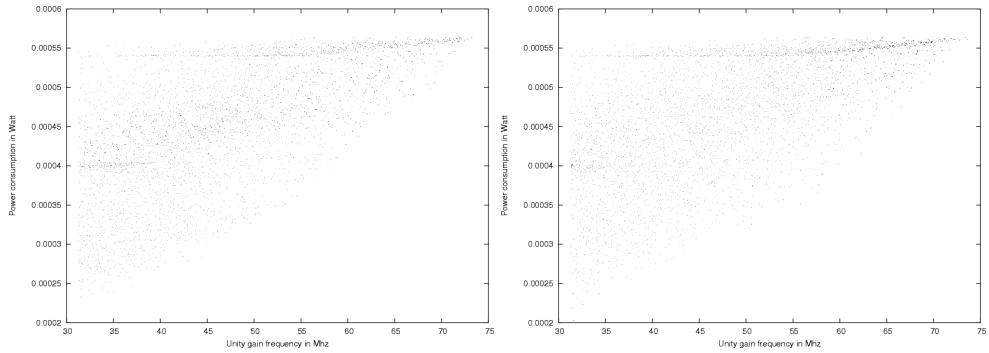



Figura 3.8: Trade-off degli algoritmi. Grafici bi-dimensionali.

### 3.3.1 Risultati MultiOb

Per quanto riguarda l'algoritmo evolutivo MultiOb la tabella 3.5 mostra i valori dei parametri più importanti:

| | |
|---|---|
| *Population size (parents)* | 1000 |
| *No. of offspring* | 500 |
| *No. of generations* | 5000 |
| *Mutation rate* | 0.1 |
| *Recombination rate 1* | 0.100000 |
| *Recombination rate 2* | 0.250000 |

Tabella 3.5: Valori dei parametri di MultiOb.

La tipologia di strategia nella selezione include i genitori, ciò permette una selezione più elitaria che non permette la deteriorazione una volta ottenuta la soluzione.

Il parametro di mutazione specifica la misura del passo in media. Si tratta della distanza media delle soluzioni risultanti dai loro rispettivi genitori, cioè la varianza delle corrispondenti variabili random. Il primo parametro di ricombinazione, specifica la probabilità di ricombinazione tra una coppia di soluzioni risultanti. Mentre il secondo parametro di ricombinazione, specifica la probabilità di ricombinazione tra una singola componente di una coppia di soluzioni risultanti.

I grafici rappresentati nelle figure 3.9 mostrano il trade-off tra triple di obiettivi. Il grafico in figura 3.10 mostra la proiezione in due dimensioni del consumo di potenza in funzione della

frequenza all'unità di guadagno, anche dai risultati ottenuti con MultiOb si ha che il consumo di potenza cresce quando cresce la frequenza all'unità di guadagno.

Tali grafici mostrano solo quegli obiettivi che soddisfano alle specifiche indicate in tabella 3.4. Si noti che, tali obiettivi sono risultati essere circa il 30% della popolazione finale di 1000 individui dopo 5000 generazioni.

Questo fatto è dovuto alle differenti metodologie utilizzate nel determinare la dominanza dei membri all'interno della popolazione: infatti, mentre l'algoritmo MultiOb seleziona attraverso un ranking a partire dalla definizione di Pareto–dominanza classica, i due algoritmi evolutivi NSGA2 e SPEA2 effettuano un calcolo della misura di *crowding–distance*, non prevista da MultiOb, per controllare la clusterizzazione.

L'algoritmo MultiOb restituisce nel fronte di Pareto una classe di circuiti che hanno un alto livello di dominanza rispetto all'insieme ammissibile delle soluzioni. Questa classe rappresenta quindi un sotto insieme ottimale stabile di soluzioni [21].

E' stato verificato, mediante simulazione Montecarlo, che l'algoritmo restituisce una soluzione robusta. Questo fatto in realtà è dovuto a un differente approccio utilizzato dall'algoritmo evolutivo MultiOb nell'estrazione: infatti, mentre gli altri algoritmi evolutivi multiobiettivo estraggono il membro della popolazione con valore migliore delle funzioni obiettivo, l'algoritmo MultiOb esegue una media tra gli individui della popolazione individuati come "migliori" che consente una stabilizzazione del risultato finale.

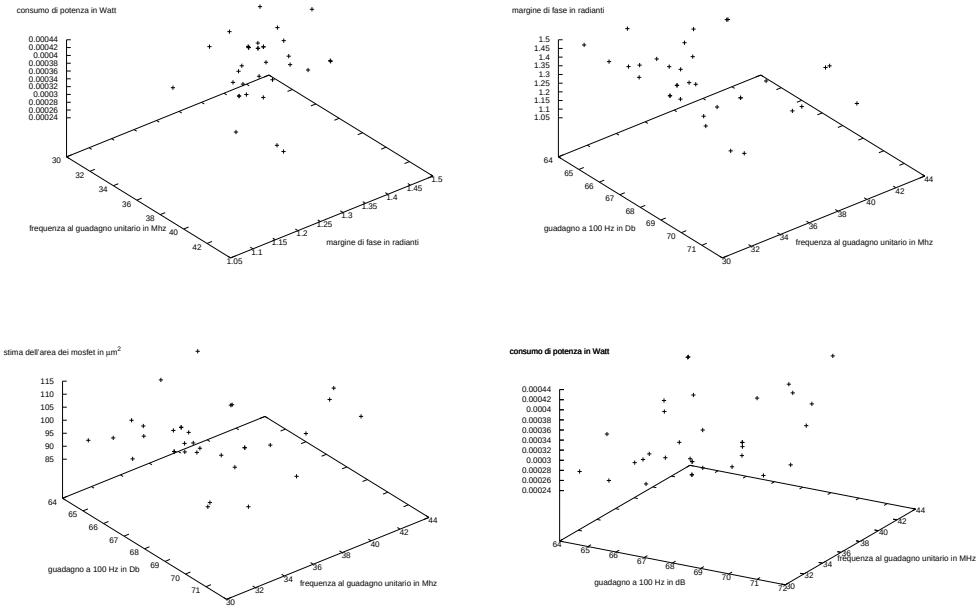



Figura 3.9: Trade-off di MultiOb.

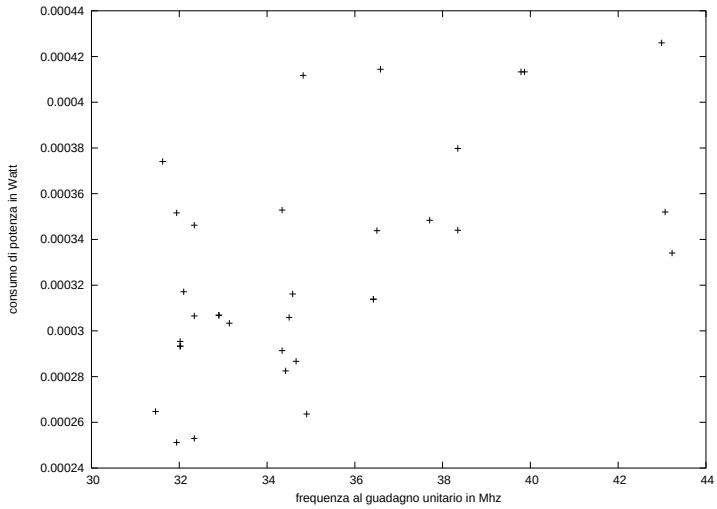



Figura 3.10: Trade-off di MultiOb

# Capitolo 4

# Worst-case Analysis

La *worst-case analysis* è comunemente utilizzata nella progettazione di circuiti integrati per garantire il soddisfacimento delle performance circuitali rispetto alle variazioni nei processi di fabbricazione nelle condizioni operative (corner) dei circuiti. L'analisi effettuata da [13] presenta un approccio che tiene conto sia delle variazioni di processo che delle fluttuazioni nelle condizioni operative. Ciò comporta condizioni del processo manifatturiero e condizioni operative di worst-case uniche e realistiche per date specifiche di circuito. Queste specifiche possono essere limiti inferiori e superiori delle performance e impattano sulla resa yield media attesa per circuito progettato.

## 4.1 Introduzione

Le fluttuazioni nei processi manifatturieri e nelle condizioni operative dei circuiti integrati (IC) determinano alcune variazioni nelle performance circuitali risultanti. Il comportamento degli IC viene comunemente simulato per le così dette condizioni del caso peggiore (worst-case) in modo da verificare che le performance siano soddisfacenti in ogni condizione. Molti metodi determinano l'insieme dei parametri di worst-case e il relativo valore della performance del worst-case tale che la probabilità delle occorrenze tra i valori di worst-case è uguale ad alcune pre-assegnate probabilità. Questo significa che gli insiemi dei parametri di worst-case e i valori di performance sono calcolati per un dato requisito minimo di yield.

Molto spesso comunque, ai progettisti di circuito vengono date delle specifiche minime di performance che devono essere soddisfatti e vogliono conoscere lo yield atteso. Un difetto

di tali metodi è che non tengono conto delle tolleranze operative. E' quindi essenziale per un designer circuitale conoscere anche le condizioni operative del caso peggiore in modo da garantire una performance di circuito soddisfacente al variare delle condizioni operative. Poichè l'esplorazione dell'intero range operativo potrebbe essere troppo dispendioso, il progettista deve essere a conoscenza delle condizioni operative e delle condizioni operative del worst-case.

Il metodo per la worst-case analysis presentato in [13] tiene conto delle tolleranze operative, così come le varianze e le correlazioni relative ai parametri del processo manifatturiero. Tale metodo fa uso di simulazioni circuitali per compensare ad una descrizione unica e realistica del comportamento circuitale del worst-case.

## 4.2 Concetti di base

Il design circuitale parametrico considera le proprietà delle performance circuitali $\boldsymbol{f}$ in funzione dei parametri circuitali $\boldsymbol{p}$ per una data topologia di circuito. Una volta che il designer ha selezionato una topologia e determinato un insieme nominale di parametri per il quale la simulazione del circuito verifica una performance nominale soddisfacente, questi ha bisogno di analizzare l'impatto del processo manifatturiero e delle tolleranze operative sul suo funzionamento.

### 4.2.1 Tolleranze di processo e operative

I parametri di processo rappresentati dal vettore $\mathbf{p}$ che sono oggetto di design si possono distinguere in due gruppi:

**parametri controllabili** parametri dei quali è possibile avere una regolazione durante le fasi di fabbricazione, ad esempio i tempi di ossidazione

**parametri non controllabili** parametri che introducono delle incertezze sul risultato finale poichè dipendono da parametri fisici e non di progettazione, ad esempio il coefficiente di crescita dell'ossido.

Entrambi i gruppi sono modellati come numeri aleatori e le loro distribuzioni congiunte di probabilità sono dedotte da misure fisiche. Tipicamente il modello per queste distribuzioni è

quello *normale* o *log-normale* sul quale si inferisce un'ipotesi di indipendenza stocastica che semplifica la formulazione.

Un'osservazione che distingue i parametri è il *livello* che essi occupano nella gerarchia che modella il sistema [6]. Sinteticamente possiamo esprimere questa gerarchia come

$$\begin{aligned} \Phi^H &= \Phi^H(\Phi^L) \\ \Phi^L &= \Phi^L(\mathbf{p}^H) \\ \mathbf{p}^H &= \mathbf{p}^H(\mathbf{p}^L) \end{aligned} \tag{4.1}$$

dove l'etichetta $L$ sta ad identificare il livello più basso dei parametri cioè quello direttamente collegato al processo (lunghezze fisiche, capacità dell'ossido, resistenze parassite ecc). Mentre $\mathbf{p}^H$ rappresenta il livello immediatamente superiore dato dalle formulazioni equivalenti, note spesso "empiricamente" (ne sono un esempio le formulazione circuitali equivalenti dei dispositivi MOS). $\Phi^L$ è il livello della rappresentazione esterna che può essere una rappresentazione dei parametri di scattering complessa, delle funzioni di trasferimento ecc. $\Phi^H$ è il livello più alto che corrisponde deve corrispondere alle misure sperimentali. Spesso i parametri non controllabili sono proprio quelli che intervengono nell'interfaccia tra i livelli e sono funzionali alla rappresentazione gerarchica. Una rappresentazione che tenga conto di queste osservazioni produce quindi una gerarchia del tipo

$$\begin{aligned} \Phi^H &= \Phi^{H,0}(\Phi^L) + \Delta F^H \\ \Phi^L &= \Phi^{L,0}(\mathbf{p}^H) + \Delta\Phi^L \\ \mathbf{p}^H &= \mathbf{p}^{H,0}(\mathbf{p}^L) + \Delta\mathbf{p}^H \\ \mathbf{p}^L &= \mathbf{p}^{L,0} + \Delta\mathbf{p}^L \end{aligned} \tag{4.2}$$

dove $\Delta\mathbf{p}^L$ corrisponde alla tolleranza sui parametri di processo, $\Delta\mathbf{p}^H$ è l'approssimazione empirica delle formule a basso livello compreso l'errore nell'estrazione dei parametri per i modelli dei dispositivi, $\Delta\Phi^L$ sono gli effetti parassiti che "disturbano" i modelli circuitali e infine $\Delta\Phi^H$ è il termine che raccoglie tutta l'incertezza sulla simulazione della performance simulata.

I circuiti integrati tipicamente si distinguono per tre tipi di parametri $\mathbf{p}$:

**d** parametri *deterministici* relativi alla geometria del circuito

**s** parametri *statistici* relativi al modello del circuito

$\theta$ parametri *operativi*

Le inevitabili tolleranze manifatturiere si riflettono in una distribuzione normale dei parametri del modello del transistor. La relativa funzione densità di probabilità *pdf*$(\mathbf{s}, \mathbf{s_0}, \mathbf{C})$ è determinata dai valori nominali $\mathbf{s_0}$ e dalla matrice di varianza-covarianza $\mathbf{C}$ dei parametri del modello del transistor. $\mathbf{s_0}$ e $\mathbf{C}$ sono determinati su base statistica e attraverso misure sul processo. Le curve di livello di uguale densità *pdf*$(\mathbf{s}, \mathbf{s_0}, \mathbf{C})$= *const* sono iper-ellissoidi con centro in $\mathbf{s_0}$ e forma determinata da $\mathbf{C}$:

$$\|\mathbf{s} - \mathbf{s_0}\|^2 = (\mathbf{s} - \mathbf{s_0})^T \cdot \mathbf{C}^{-1} \cdot (\mathbf{s} - \mathbf{s_0}) = \beta^2 \tag{4.3}$$

Un iper-ellissoide determinato da un valore costante $\beta$ rappresenta il così detto corpo di tolleranza. La notazione $\|\cdot\|$ denota una norma ellissoidale in accordo alla 4.3.

I parametri geometrici di un circuito progettabile sono parametri deterministici, poichè le loro fluttuazioni statistiche sono raggruppate nei parametri di modello riduzione larghezza/lunghezza.

I parametri operativi sono soggetti a fluttuazioni. Il funzionamento del circuito deve essere garantito per specifici intervalli di tolleranza dei parametri operativi con limiti inferiori $\theta_{\mathbf{L}}$ e limiti superiori $\theta_{\mathbf{U}}$:

$$\theta_{\mathbf{L}} \leq \theta \leq \theta_{\mathbf{U}} \tag{4.4}$$

## 4.3 Yield circuitale

Un circuito è classificato *accettabile* nelle performance se tutte le specifiche sono rispettate. Il termine resa (*yield*) nel contesto dell'industria microelettronica individua il rapporto tra il numero di chip che hanno una performance accettabile sul totale di quelli prodotti.

$$\text{Yield} = \frac{\#\text{chip accettabili}}{\#\text{chip prodotti}} \tag{4.5}$$

Le cause che rendono *inaccettabili* le performance di un circuito sono classificate in due categorie di *perturbazioni*: **locale** e **globale**.

**Perturbazione locale** causata dalla rottura della struttura cristallina del silicio che determina in genere il malfunzionamento di un singolo chip del wafer.

**Perturbazione globale** causata da imprecisioni durante i processi produttivi quali disallineamento delle maschere, variazioni di temperatura, variazioni delle dosi di impianto.

A differenza della perturbazione locale, quella globale coinvolge tutti i chip del wafer anche in misura diversa nelle diverse regioni della fetta di wafer. Gli effetti di questa perturbazione sono in genere il mancato raggiungimento delle performance richieste: diminuzione della frequenza di lavoro, aumento del consumo di potenza ecc.

Questa classificazione pratica è in realtà una **semplificazione** poichè le cause che determinano la perdita di performance sono dovute invece a fenomeni fisici trascurati. Ad esempio effetti parassiti come accoppiamento elettromagnetico tra elementi, dissipazioni, dispersioni ecc. Tuttavia l'approccio sopra esposto semplifica la trattazione poichè introduce un concetto di errore statistico che permette di utilizzare concetti e strumenti tipici delle metodologie statistiche.

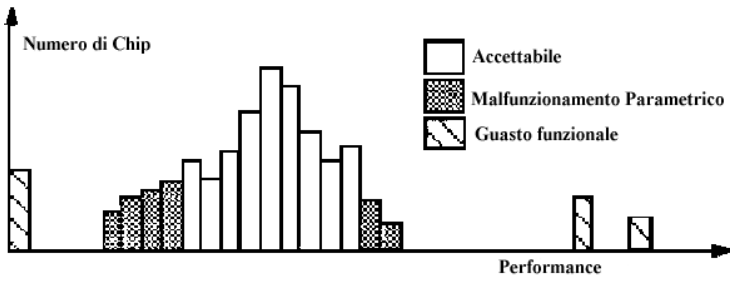



Figura 4.1: Distribuzione delle performance nella fabbricazione di circuiti integrati

Lo *yield* $Y_i$ visto come la percentuale dei circuiti manufatti che soddisfano una specifica come limite inferiore di performance $L_i$ oppure limite superiore di performance $U_i$. Lo yield è uguale alla probabilità dell'occorrenza di una proprietà di performance $f_i$ entro i valori dei limiti superiore/inferiore. $Y_i$ è quindi determinato dal seguente integrale multi-dimensionale:

$$Y_i = prob\{f_i \geq L_i / f_i \leq U_i\} = \int pdf(s, s_0, C) \cdot ds \tag{4.6}$$

Il metodo mostrato per la worst-case analysis richiede una mappatura del valore di yield $Y_i$ secondo la 4.6 all'interno del corpo di tolleranza definito da $\beta_i$ secondo la relazione 4.3. Esistono due metodi fondamentali per mettere in relazione $Y_i$ e $\beta_i$ tra loro.

**Metodo I** Definisce la mappatura $Y_i \leftrightarrow \beta_i$ tale che la percentuale dei circuiti all'interno dell'iper-ellissoide definito in 4.3 è uguale ad uno specifico valore di yield.

$$\beta_i \mapsto Y_i : Y_i = \varphi(\beta_i) = \int_{\{s|\|s-s_0\| \leq \beta_i\}} pdf(s, s_0, C) \cdot ds \tag{4.7}$$

$$Y_i \mapsto \beta_i : \beta_i = \varphi_{-1}(Y_i) \tag{4.8}$$

Deve essere calcolato un integrale multi-dimensionale per mappare $\beta_i$ su $Y_i$ e vice versa.

Il Metodo I spesso conduce ad un limite inferiore della stima dello yield. Comunque, questo limite inferiore è spesso troppo pessimistico. La figura 4.2 a sinistra illustra il Metodo I per 2 parametri statistici e una proprietà di performance $f_i$ con un limite superiore $U_i$.

I circuiti che soddisfano il limite superiore giacciono all'interno della regione di accettazione determinata da $f_i < U_i$. L'iper-ellissoide massimale secondo la 4.3 all'interno della regione di accettazione è definita da $\beta = 2.0$. Con lo yield definito dal Metodo I, viene ottenuto un valore del 67%. Si può notare come questo risultato sia molto pessimistico poichè l'area della regione di accettazione non-ombreggiata è trascurata.

**Metodo II** Definisce una mappatura $Y_i \leftrightarrow \beta_i$ tale che la percentuale dei circuiti su un lato dell'iper-plano tangente del corpo di tolleranza 4.3 è uguale ad un dato valore di yield. Si può provare che la mappatura è definita dal seguente integrale uni-dimensionale.

$$\beta_i \mapsto Y_i : Y_i = cdf(\beta_i) = \int_{-\infty}^{\beta_i} \frac{1}{\sqrt{2\pi}} e^{(-\beta^2/2)} d\beta \tag{4.9}$$

$$Y_i \mapsto \beta_i : \beta_i = cdf^{-1}(Y_i) \tag{4.10}$$

$cdf$ è la funzione di distribuzione cumulativa della $pdf$ unidimensionale normale. $cdf$ e $cdf^{-1}$ sono date dalle tabelle statistiche standard. Il Metodo II di solito conduce a condizioni più realistiche di worst-case rispetto al Metodo I. Ad esempio in figura 4.2 a destra, il Metodo II risulta in un valore più realistico di yield del 97%.

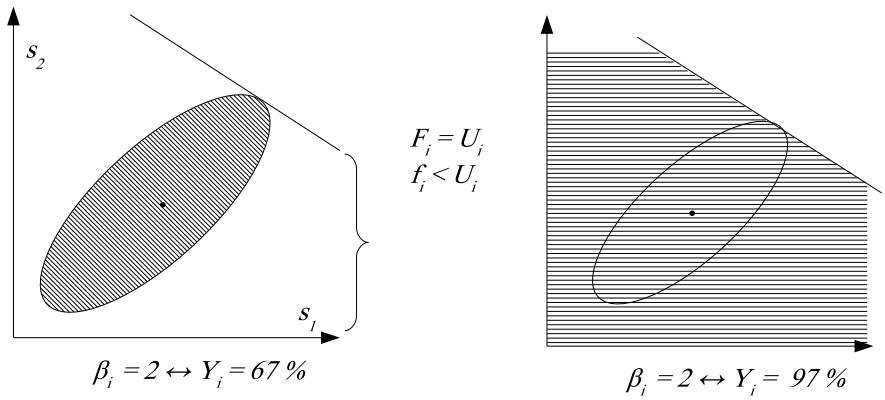



Figura 4.2: Valori di yield riferiti ad un corpo di tolleranza determinato da Metodo I (a sinistra) e Metodo II (a destra)

### 4.3.1 Massimizzazione dello yield

In questa sezione saranno date le definizioni formali del problema della *massimizzazione parametrica dello yield*. Sia $\mathbf{\Phi} = (\phi_1, \phi_2, \ldots, \phi_m)$ il vettore delle performance determinate dal vettore di parametri $\mathbf{p} = (p_1, p_2, \ldots, p_n)$.

> $\mathbf{\Phi}$ e $\mathbf{p}$ sono considerati *numeri aleatori* per il problema della massimizzazione della resa e la relazione che intercorre tra questi due vettori è sintetizzata come $\mathbf{\Phi} = \mathbf{\Phi}(\mathbf{p})$.

Formalmente il problema ha la stessa formulazione sia quando viene visto in termini di parametri di processo, sia quando viene visto nei termini di resa circuitale. In questo senso d'ora in avanti ci si riferirà al problema nei termini di resa circuitale sottintendendo la formulazione in termini di parametri di processo per i dispositivi microelettronici.

Un circuito prodotto è considerato *accettabile* se tutte le performance cadono in range determinati da target di accettabilità. In simboli

$$\phi_k^L \leq \phi_k \leq \phi_k^U, \quad k = 1, \ldots, m. \tag{4.11}$$

Le disuguaglianze 4.11 definiscono una *regione di accettabilità* $\mathcal{A}_{\mathbf{\Phi}}$ nello spazio cartesiano a $m$ *dimensioni* delle performance come

$$\mathcal{A}_{\Phi} = \{\mathbf{\Phi} \mid \phi_k^L \leq \phi_k \leq \phi_k^U, \quad k = 1, \ldots, m\} \tag{4.12}$$

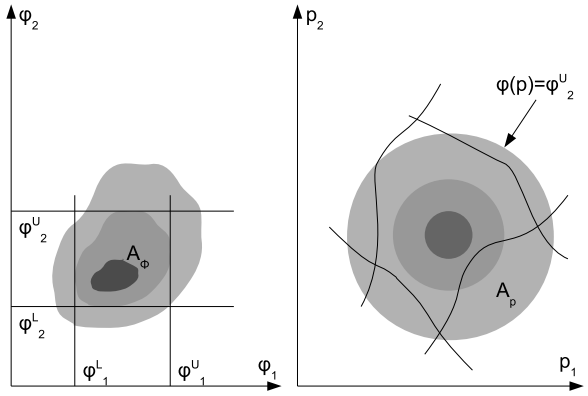



Figura 4.3: Regione di accettabilità in relazione alle performance (a destra) e ai parametri (a sinistra)

A partire dalle precedenti definizioni si definisce *resa di design* la probabilità che un circuito prodotto abbia una performance accettabile cioè

$$Y = Prob\{\mathbf{\Phi} \in \mathcal{A}_{\mathbf{\Phi}}\} = \int_{\mathcal{A}_{\mathbf{\Phi}}} f(\mathbf{\Phi}) d\mathbf{\Phi} \tag{4.13}$$

dove $f$ è la densità di probabilità congiunta delle performance $\mathbf{\Phi}$.

La massimizzazione della resa è elaborata attraverso la ricerca di un vettore $\overline{\mathbf{p}}$ nominale nello spazio dei parametri che influisca sui momenti della distribuzione casuale dei parametri $\mathbf{p}$ e conseguentemente sulle performance $\Phi$, in modo che la maggior parte della distribuzione di probabilità delle performance giaccia nella regione $\mathcal{A}_{\mathbf{\Phi}}$. Ridefinendo la regione di accettabilità in termini di parametri si ha:

$$\mathcal{A}_{\mathbf{p}} = \{\mathbf{p} \mid \phi_k^L \leq \phi_k(\mathbf{p}) \leq \phi_k^U, \;\; k = 1, \ldots, m\} \tag{4.14}$$

Formalmente quindi la resa di design 4.13 viene riformulata come:

$$Y = Prob\{\mathbf{p} \in \mathcal{A}_{\mathbf{p}}\} = \int_{\mathcal{A}_{\mathbf{p}}} f(\mathbf{p}, \overline{\mathbf{p}}) d\mathbf{p} \tag{4.15}$$

Il problema risulta quindi essere la massimizzazione della probabilità di resa nello spazio dei parametri $\mathbf{p}$ in termini di "*centramento*" rispetto ad un set di parametri nominali $\overline{\mathbf{p}}$

$$\max_{\overline{\mathbf{p}}} \left[ Y = \int_{\mathcal{A}_{\mathbf{p}}} f(\mathbf{p}, \overline{\mathbf{p}}) d\mathbf{p} \right] \tag{4.16}$$

Una formulazione alternativa al "centramento" è quella che definisce il problema in uno spazio delle *perturbazioni*. Lo spazio delle perturbazioni è uno spazio cartesiano di variabili casuali indipendenti $\xi$. Lo spazio di accettabilità viene quindi riformulato come:

$$\mathcal{A}_\xi(\overline{\mathbf{p}}) = \{\xi \mid \phi_k^L \leq \phi_k(\overline{\mathbf{p}}, \xi) \leq \phi_k^U, \;\; k = 1, \ldots, m\} \tag{4.17}$$

cioè lo spazio dato da tutte le combinazioni delle perturbazioni alteranti i processi produttivi per uno specifico vettore $\overline{\mathbf{p}}$ di parametri nominali che non rendono *inaccettabili* le performance del circuito.

Diversamente dalla formulazione di "centramento", la regione di accettabilità in questo caso si ottiene come mappa inversa di $\xi \rightarrow \mathbf{\Phi}$ che dipende dalla scelta dei valori nominali $\overline{\mathbf{p}}$. La formulazione della probabilità di resa viene quindi espressa in simboli come:

$$Y = Prob\{\xi \in \mathcal{A}_\xi(\overline{\mathbf{p}})\} = \int_{\mathcal{A}_\xi(\overline{\mathbf{p}})} f(\xi) d\xi \tag{4.18}$$

Il problema di ottimizzazione si esprime quindi nel seguente modo:

$$\max_{\overline{\mathbf{p}}} \left[ Y = \int_{\mathcal{A}_\xi(\overline{\mathbf{p}})} f(\xi) d\xi \right] \tag{4.19}$$

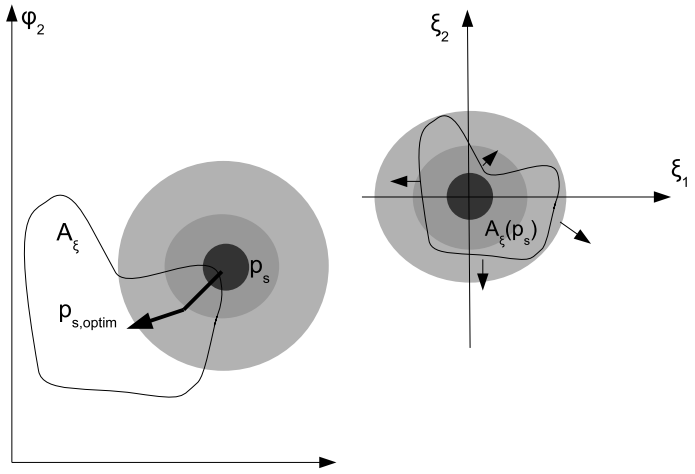



Figura 4.4: Rappresentazione del problema della massimizzazione della resa in termini di centramento (a destra) e di perturbazione (a sinistra).

In generale le formulazioni 4.16 e 4.19 non sono equivalenti, poichè nella prima si suppone che la scelta dei parametri influenzi solo le probabilità congiunte mentre nella seconda influen-

zi solamente la regione di accettabilità (definita in termini di perturbazioni). In questo senso formulazioni più generali possono essere modellate con un approccio misto come:

$$\max_{\overline{\mathbf{p}}} \left[ Y = \int_{\mathcal{A}_\xi(\overline{\mathbf{p}})} f(\xi, \overline{\mathbf{p}}) d\xi \right] \tag{4.20}$$

Riassumendo una formulazione della resa in termini di *tolleranze* conduce ad un problema di tipo 4.16 mentre una formulazione in termini di specifica conduce ad un problema di tipo 4.19.

### 4.3.2 Worst-case optimization

Il trattamento del caso peggiore è implementato in diverse varianti in molte metodologie. Questo tipo di approccio è basato su una strategia conservativa che stabilisce il *caso peggiore* come l'eventualità dalla quale allontanarsi nel pianificare l'ottimizzazione. Una metodologia di questo tipo è rintracciabile in [2] e usa una funzione di resa quadratica costruita con una regressione su un sottoinsieme dei parametri che costituisce un macromodello per il problema. L'ipotesi di base è che i parametri per il modello siano costituiti dalla somma tra il loro *valore nominale* e una perturbazione con **media zero e varianze fissate**. Il problema viene quindi formulato in termini di "*maxmin*" come:

$$\max_{\mathbf{p}_0 \in \mathcal{A}_p} \min_{Q} r(\mathbf{p}_0) \tag{4.21}$$

dove $Q$ sono gli iperpiani che determinano la regione di accettabilità $\mathcal{A}_p$. Questa metodologia utilizza la *quadratic programming* per risolvere il problema 4.21.

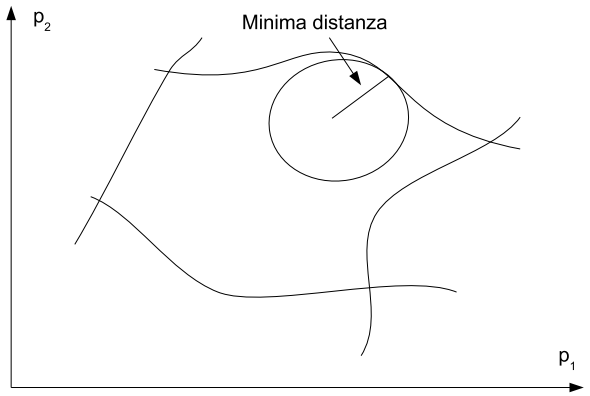



Figura 4.5: La figura illustra il centramento come problema di maxmin distanza

In [5] viene esplicitato il legame tra *worst–case* e resa della performance $i$-esima attraverso l'approssimazione:

$$Y_i = \int_{\mathcal{A}_{\mathbf{\Phi_i}}} f(\mathbf{\Phi_i}) d\mathbf{\Phi_i} \simeq \int_{-\infty}^{\beta_{w,i}} \frac{1}{\sqrt{2\pi}} \cdot \exp(-\frac{t^2}{2}) \cdot dt \tag{4.22}$$

dove $\beta_{w,i}$ è la norma worst–case del parametro $i$-esimo rispetto al valore nominale calcolata attraverso l'inversa della matrice di correlazione come:

$$\beta_{w,i} = (\mathbf{p}_{w,i} - \mathbf{p}_0)^T \cdot \mathbf{C}^{-1} \cdot (\mathbf{p}_{w,i} - \mathbf{p}_0) \tag{4.23}$$

Va osservato che queste formulazioni possono spesso condurre a problemi con vincoli troppo rigidi da soddisfare e che determinano soluzioni lontane da quelle desiderate. Una articolata implementazione di questa metodologia può essere rintracciata in [4], dove gli autori rivolgono l'attenzione alla soluzione alle problematiche relative alla formulazione del problema nonlineare focalizzando l'attenzione su:

- linearizzazione della specifica delle performance sul punto di worst case dei parametri statistici;
- linearizzazione della regione di ammissibilità attraverso la definizione di una "trust" region delle performance rispetto ai parametri di design;
- una robusta metodologia di ricerca sulle coordinate di design.

# Capitolo 5

# Insiemi fuzzy e Teoria della possibilità

Questo capitolo introduce i concetti di insieme fuzzy e teoria della possibilità che permetteranno l'estensione della worst–case analysis al caso possibilistico e saranno i principali elementi a cui si farà ricorso nella metodologia Possibilistic Worst–Case Distance (PWCD).

La *teoria degli insiemi fuzzy* può utilizzare un numero di dati disponibili limitato e tenere conto di scenari del caso peggiore. La teoria degli insiemi fuzzy è così in grado di compensare l'incertezza dal momento che questa è modellata basandosi su opinioni e assunzioni soggettive. Al contrario, i metodi probabilistici richiedono un gran numero di dati e i risultati ottenuti sono, in alcuni casi, molto sensibili, sia all'accuratezza dei dati stessi, sia alle assunzioni fatte durante la modellazione del processo.

La teoria degli insiemi fuzzy è stata introdotta da Zadeh come strumento matematico per la modellazione quantitativa dell'incertezza e fa uso dei numeri fuzzy per rappresentare l'incertezza dei parametri del problema. Il progettista ha bisogno di specificare il range di incertezza e la funzione di appartenenza che denota la possibilità di occorrenze di un elemento in un range specificato per rappresentare un parametro incerto come numero fuzzy. Le funzioni di appartenenza generalmente sono costruite in maniera soggettiva basata sull'opinione di un esperto.

La *teoria della possibilità*, anch'essa introdotta da Zadeh [24], è una alternativa alla teoria della probabilità nelle situazioni in cui si tiene conto dell'incertezza. La misura di possibilità è una misura fuzzy, e le distribuzioni di possibilità sono in corrispondenza uno a uno con gli insiemi fuzzy. Oltre alla misura di possibilità, c'è anche la sua duale misura di necessità: un evento è necessario se il suo contrario non è possibile.

## 5.1 Teoria degli insiemi fuzzy

La teoria degli insiemi fuzzy presenta una metodologia per la modellazione matematica dell'incertezza. Al contrario della teoria classica degli insiemi dove esiste una transizione netta tra appartenenza e non-appartenenza all'insieme, la teoria degli insiemi fuzzy fa uso delle funzioni di appartenenza per denotare il grado per il quale un elemento appartiene ad un insieme fuzzy. Una *funzione di appartenenza* assegna un grado di appartenenza tra 0 ed 1 a ciascun elemento dell'universo come segue:

$$M(x) : X \rightarrow [0, 1] \tag{5.1}$$

Nella 5.1 **M** denota una funzione di appartenenza che mappa gli elementi dell'insieme universale **X** sull'intervallo reale [0, 1].

Gli insiemi fuzzy sono rappresentati numericamente facendo uso degli $\alpha$ livelli. Un $\alpha$ *livello* è definito come un intervallo reale dove la funzione di appartenenza è più grande di un dato valore, $\alpha$, e può essere scritto matematicamente per un generico insieme fuzzy **B** come segue [16]:

$$^{\alpha}B = x|B(x) \geq \alpha \tag{5.2}$$

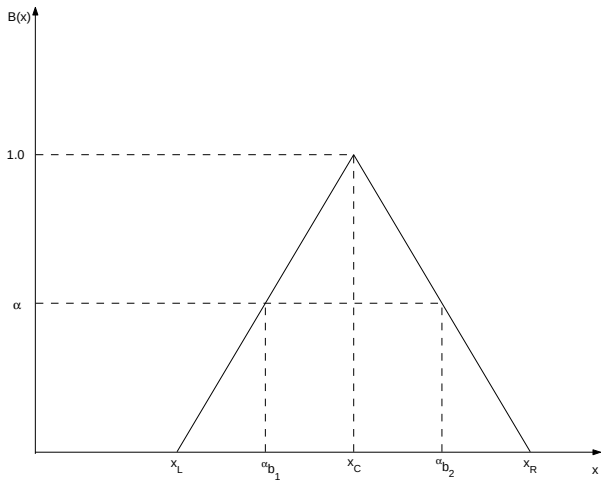



Figura 5.1: Rappresenta un grafico per la 5.2, dove si assume che **B** ha una funzione di appartenenza triangolare e simmetrica, e mostra i punti estremi $^{\alpha}b_1$ ed $^{\alpha}b_2$ di un generico $\alpha$ livello.

Un numero fuzzy è definito come un insieme fuzzy che sia normale e convesso. Un insieme fuzzy normale ha un massimo della funzione di appartenenza ad 1, mentre tutti i possibili $\alpha$ livelli sono convessi per insiemi fuzzy convessi. L'insieme fuzzy **B** mostrato in figura 5.1 è quindi un numero fuzzy. In effetti, la funzione di appartenenza triangolare e simmetrica è la più utilizzata per rappresentare numeri fuzzy, principalmente a causa della sua semplicità.

Una funzione fuzzy $\mathbf{Y}$ è una funzione di variabili fuzzy $\mathbf{X_i}$ e può essere scritta, nel caso di *n* variabili fuzzy, come:

$$\mathbf{Y} = \mathbf{Y}(\mathbf{X_1}, \mathbf{X_2}, ..., \mathbf{X_n}) \tag{5.3}$$

Klir e Yuan [16] provarono le seguenti proprietà di una funzione fuzzy:

1. Se tutte le variabili fuzzy di una funzione fuzzy sono numeri fuzzy continui, allora la funzione fuzzy stessa è anche un numero fuzzy continuo.

2. Se tutte le variabili fuzzy di una funzione fuzzy sono numeri fuzzy, l' $\alpha$ livello della funzione fuzzy ${}^{\alpha}Y$ può essere scritta in termini degli $\alpha$ livelli relativi a tali variabili fuzzy ${}^{\alpha}X_i$ come segue:

$$\begin{aligned} &{}^{\alpha}Y = {}^{\alpha}\, Y({}^{\alpha}X_1, \ldots, {}^{\alpha}\, X_n) = \\ &\left[\min_{{}^{\alpha}R} \left[{}^{\alpha}Y({}^{\alpha}X_1, \ldots, {}^{\alpha}\, X_n)\right], \max_{{}^{\alpha}R} \left[{}^{\alpha}Y({}^{\alpha}X_1, \ldots, {}^{\alpha}\, X_n)\right]\right] \end{aligned} \tag{5.4}$$

   Dove ${}^{\alpha}R$ denota l'ipercubo n-dimensionale, formato dagli $\alpha$ livelli degli *n* numeri fuzzy.

Basandosi su queste proprietà di una funzione fuzzy, Dong e Shah(1987) introdusse il metodo dei vertici per valutare i limiti inferiore e superiore di ${}^{\alpha}Y$ quando tutte le variabili fuzzy di **Y** sono numeri fuzzy. Questo metodo richiede la valutazione della funzione fuzzy nei $2^n$ vertici del rettangolo n-dimensionale, formato dagli $\alpha$ livelli delle *n* variabili fuzzy. In aggiunta, bisogna controllare anche i punti interni di estremo globale. Questo metodo richiede un gran numero di valutazione di funzioni ed è computazionalmente dispendioso.

Per calcolare la possibilità di fallimento si deve confrontare un numero crisp con un numero fuzzy. Un modo per fare ciò è utilizzare la teoria della possibilità introdotta da Zadeh(1978) e sviluppata ed estesa successivamente da Dubois e Prade.

## 5.2 Teoria della possibilità

La teoria della possibilità è una teoria più generale di quella della probabilità nella quale si tiene conto di incertezza. Il Professor Lotfi Zadeh [24] per primo introdusse la teoria della possibilità nel 1978 come estensione della sua teoria di insiemi fuzzy e logica fuzzy. D. Dubois and H. Prade [11, 12] in seguito contribuirono allo sviluppo di tale teoria. Nei primo anni '50 l'economista G.L.S. Shackle propose l'algebra min/max per descrivere i gradi di potenziale sorpresa.

### 5.2.1 Possibilità

Sia $X$ un insieme arbitrario, qui di seguito viene introdotto il concetto di misura di possibilità in generale:

**Definizione 5.2.1** *Una **misura di possibilità** $\Pi$ su $X$ è una funzione $\Pi : \mathcal{P}(X) \rightarrow [0,1]$ tale che:*

1. $\Pi(\emptyset) = 0$
2. $\Pi(X) = 1$
3. *Per ogni collezione $(A_i)_i \in I$ di sottoinsiemi di $X$,*

$$\Pi(\bigcup_{i \in I} A_i) = \sup_{i \in I} \Pi(A_i) \tag{5.5}$$

La prima proprietà può essere interpretata come l'assunzione che $X$ sia una descrizione esaustiva dei futuri stati del mondo, poichè significa che non è data alcuna possibilità agli elementi al di fuori di $X$. La seconda proprietà può essere interpretata come l'assunzione che l'evidenza dalla quale $\Pi$ è stata costruita è libera da ogni contraddizione. Tecnicamente ciò implica che, esiste almeno un elemento in $X$ con possibilità pari ad 1. L'ultima proprietà corrisponde all'assioma di additività della teoria della probabilità.

Concetto associato alla misura di possibilità è la distribuzione di possibilità.

**Definizione 5.2.2** *Una **distribuzione di possibilità** è una funzione $\pi : X \rightarrow [0,1]$ tale che* $\sup_{x \in X} \pi(x) = 1$.

Quello che si vede facilmente è che c'è una corrispondenza uno a uno tra misure e distribuzioni. Vale infatti la seguente:

**Proposizione 5.2.1** *Sia $M_\Pi$ l'insieme delle misure di possibilità su $X$ e sia $D_\pi$ l'insieme delle distribuzioni di possibilità su X. Allora $M_\Pi$ e $D_\pi$ sono corrispondenza biunivoca.*

#### Proprietà delle possibilità

Qui di seguito vengono esposte alcune proprietà fondamentali delle possibilità.

**Proposizione 5.2.2** *Sia $\Pi$ una misura di possibilità su $X$, allora valgono:*

- $\forall A, B \in \mathcal{P}(X), \Pi(A \cup B) = \max\{\Pi(A), \Pi(B)\}$,
- $\forall A, B \in \mathcal{P}(X), \Pi(A \cap B) \leq \min\{\Pi(A), \Pi(B)\}$.

**Proposizione 5.2.3** *Sia $\Pi$ una misura di possibilità su $X$, allora vale:*

$$\forall A \in \mathcal{P}(X), \max\{\Pi(A), \Pi(\overline{A})\} = 1$$

Dalla proposizione sopra segue immediatamente che:

$$\forall A \in \mathcal{P}(X), \Pi(A) + \Pi(\overline{A}) \ \geq 1.$$

Nel caso di possibilità definite su insiemi $X$ di cardinalità finita, $|X| < +\infty, X = X_1, \ldots, X_k$. In questo caso dare una distribuzione di possibilità equivale a dare un vettore $\pi = \pi_1, \ldots, \pi_k \in \mathbb{R}^k$, tale che $\max_{1 \leq i \leq k} \pi_i = 1$. Ciò significa che $\exists i^* \in \{1, \ldots, k\}$ tale che $\pi_{i^*} = 1$.

### 5.2.2 Necessità

Mentre la teoria della probabilità utilizza un solo valore per indicare se un evento accade, La teoria della possibilità utilizza due concetti, la *misura di possibilità* e la *misura di necessità*. Verrà mostrato successivamente come il concetto di misura di necessità sia il concetto duale rispetto a quello di possibilità.

**Definizione 5.2.3** *Una **misura di necessità** $N$ su $X$ è una funzione $N : \mathcal{P}(X) \to [0, 1]$ tale che:*

1. $N(\emptyset) = 0$

2. $N(X) = 1$

3. *Per ogni collezione* $(A_i)_i \in I$ *di sottoinsiemi di* $X$*,*

$$N(\bigcap_{i \in I} A_i) = \inf_{i \in I} N(A_i) \tag{5.6}$$

Vale dunque il seguente:

**Teorema 5.2.1** *Sia* $\Pi$ *una misura di possibilità. Allora la funzione d'insieme* $N$ *definita da:*

$$\forall A \in \mathcal{P}(X), \Pi(A) + N(\overline{A}) = 1.$$

*è una misura di necessità.*

**Proprietà delle necessità**

Questo paragrafo mostra una serie di proprietà delle necessità:

**Proposizione 5.2.4** *Sia* $N$ *una misura di necessità su* $X$*, allora valgono:*

- $\forall A, B \in \mathcal{P}(X), N(A \cup B) = \min\{N(A), N(B)\}$,
- $\forall A, B \in \mathcal{P}(X), N(A \cap B) \geq \max\{N(A), N(B)\}$.

Questi risultati seguono subito dalla dualità con le possibilità. Un altra proprietà che segue facilmente dalla dualità con le possibilità è la seguente:

**Proposizione 5.2.5** *Sia* $N$ *una misura di necessità su* $X$*, allora vale:*

$$\forall A \in \mathcal{P}(X), \min\{N(A), N(\overline{A})\} = 0$$

### 5.2.3 Rapporto tra possibilità e necessità

Vengono esposte ora alcune proprietà che concernono la connessione tra i concetti di possibilità e di necessità, e che garantiscono che questa teoria sia valida da un punto di vista logico e descrittivo. La relazione fondamentale (al di là della dualità) che lega tra loro questi due concetti è il seguente teorema a cui viene dato il nome di teorema debole.

**Teorema 5.2.2 (teorema debole)** *Sia* $\Pi$ *una misura di possibilità su* $X$ *e sia* $N$ *la corrispondente misura duale di necessità. Allora vale:*

$$\forall A \in \mathcal{P}(X), N(A) \leq \Pi(A).$$

L'altra proprietà fondamentale che lega possibilità e necessità, e che esplica in maniera più chiara il legame concettuale vigente tra questi due concetti, è la seguente:

**Proposizione 5.2.6** *Sia $\Pi$ una misura di possibilità su $X$ e sia $N$ la corrispondente misura duale di necessità. Allora valgono $\forall A \in \mathcal{P}(X)$:*

- $\Pi(A) < 1 \Rightarrow N(A) = 0$ *(condizione di contingenza debole)*
- $N(A) > 0 \Rightarrow \Pi(A) = 1$ *(condizione di contingenza forte)*

Da qui vediamo che un evento, per essere un poco necessario, deve essere pienamente possibile. Se non è pienamente possibile allora è contingente e quindi non necessario. In particolare, noi possiamo pensare di associare ad ogni evento A un intervallo $[N(A), \Pi(A)]$, che è ben definito grazie al teorema debole. La proposizione chiarifica quali tipi di intervalli sono permessi, come si vede nel seguente:

**Corollario 5.2.1 (Teorema forte)** *Sia $\Pi$ una misura di possibilità su $X$, e sia $N$ la corrispondente misura duale di necessità. L'intervallo $[N(A), \Pi(A)]$ è di uno dei tipi seguenti:*

- $[0, 0]$ *(impossibilità)*
- $[1, 1]$ *(certezza)*
- $[0, 1]$ *(ignoranza)*
- $[\alpha, 1]$ *con* $0 < \alpha < 1$ *(contingenza forte)*
- $[0, \alpha]$ *con* $0 < \alpha < 1$ *(contingenza debole)*

## 5.3 Rapporto tra probabilità e possibilità

Ci si potrebbe ora chiedere che legame ci sia, in generale, tra una probabilità e una possibilità. I due concetti sono chiaramente distinti, come ci suggerisce il linguaggio comune. Siamo soliti dire, infatti: "è possibile ma poco probabile che un certo evento accada" oppure "è improbabile ma non impossibile che un certo evento accada". Il concetto di possibilità è più debole di quello di probabilità, nel senso che ha meno implicazioni. Questo suggerisce di adottare il seguente principio.

**Definizione 5.3.1 (Principio di consistenza probabilità-possibilità.)** *Sia $P$ una probabilità su $X$ e sia $\Pi$ una possibilità su X. Diremo che $P$ è consistente con $\Pi$ se:*

$$\forall A \in \mathcal{P}(X), P(A) \leq \Pi(A) \tag{5.7}$$

La dualità tra possibilità e necessità induce una condizione di consistenza tra probabilità e necessità. Anche questa definizione è estremamente ragionevole, in quanto un evento necessario è sicuramente probabile.

**Osservazione 5.3.1** *Sia $\Pi$ una misura di possibilità su $X$. Allora:*

$$\forall A \in \mathcal{P}(X), P(A) \leq \Pi(A) \Rightarrow \forall A \in \mathcal{P}(X), P(A) \geq N(A) \tag{5.8}$$

*ove $N$ è la misura di necessità duale di $\Pi$.*

## 5.4 Possibilità e misure fuzzy

Le misure di possibilità sono delle misure fuzzy. In effetti queste ultime sono chiamate anche distribuzioni di fiducia, nome che sembra più consono all'interno di teorie che studiano dei modelli per la conoscenza parziale. Vediamo meglio questo fatto: le misure fuzzy, sono delle funzioni da una classe arbitraria $\mathcal{A}$ di sottoinsiemi di $X$ in $\mathbb{R}^+ = [0, +\infty]$ che soddisfano alle condizioni di monotonia e di misura nulla dell'insieme vuoto. E' quindi chiaro che le possibilità e le necessità, essendo funzioni monotone che attribuiscono misura nulla all'insieme vuoto, sono delle misure fuzzy.

Una possibile generalizzazione delle possibilità ad eventi fuzzy. L'idea di fondo è che non sempre è possibile descrivere gli eventi in modo preciso, e anzi spesso essi sono intrinsecamente fuzzy (per esempio quando mi chiedo quale sia la possibilità che una certa persona sia ricca).

### 5.4.1 Rapporto tra possibilità e insiemi fuzzy

Denotiamo con $x$ una variabile che prende i suoi valori su di un insieme $X$, e l'informazione concernente detta variabile sia espressa da una proposizione fuzzy '$x$ è $F$', dove $F$ è un sottoinsieme fuzzy di $X$. Ora, è naturale interpretare il grado si appartenenza $F(y)$, per ogni $y \in X$, come il grado di possibilità che $x = y$. In questo modo si riesce a costruire una distribuzione di possibilità $\pi_F$ , definendola come:

$$\pi_F(x) = F(x)$$

per ogni $x \in X$.

Questo tipo di ragionamento funziona bene se l'insieme fuzzy da cui si parte è normale (cioè $sup_{x \in X} F(x) = 1$). In questo modo, infatti, si origina una distribuzione di possibilità secondo la definizione data precedentemente. Poichè la maggior parte degli insiemi fuzzy non è normale, quest'interpretazione sembra essere debole, a meno che non si riescano ad inquadrare nell'ambito delle possibilità anche distribuzioni non normalizzate.

# Capitolo 6

# Ottimizzazione OTA: Possibilistic Worst-Case Distance

Il metodo di ottimizzazione presentato in questo capitolo si inspira a [23] e ad esso è stato dato il nome di *Possibilistic Worst-Case Distance (PWCD)*. Tale metodologia è stata testata sull'applicazione del sizing di dispositivi MOS di una rete di circuito rappresentante un Operational Transconductance Amplifier a due stadi (two-stage OTA) descritto nel capitolo 1.

Per modellare l'incertezza dei parametri di circuito è stata utilizzata la teoria degli insiemi fuzzy. Una linearizzazione sulla risposta del circuito al variare dei parametri incerti viene utilizzata come approssimazione accurata sull'intero spazio di design, principalmente per ridurre il costo computazionale associato al design in condizioni di incertezza. Mediante i coefficienti ottenuti con l'operazione di linearizzazione, viene effettuata la fuzzificazione delle risposte del circuito. Infine, la procedura calcola la misura della possibilità di fallimento delle performance circuitali rispetto a determinate specifiche e questa è stata minimizzata.

Il caso di studio mostrerà come l'approccio possibilistico, benchè meno accurato nella stima indiretta dello yield della Statistical Worst-Case Analysis, può individuare ugualmente un design ottimale in termini di yield. Inoltre, la metodologia possibilistica, permette di sviluppare i calcoli senza far ricorso a stringenti ipotesi statistiche e a complesse analisi di sensibilità.

## 6.1 Introduzione

Il riscalamento della tecnologia CMOS è accelerato negli ultimi anni e probabilmente proseguirà verso il regime di 10nm. L'esplorazione sul design circuitale avanzato dovrebbe incominciare prima che le future tecnologie siano pienamente sviluppate in modo da risolvere le sfide di sviluppo imposte dalla tecnologia nanoscalare CMOS. Particolari esempi di queste sfide emergenti sono correnti parassite, variazioni di processo, e affidabilità del transistor. Per i primi design circuitali, è critico avere modelli di MOSFET predittivi che siano ragionevolmente accurati, riscalabili e che catturino correttamente questi nuovi effetti fisici. Per predire le future caratteristiche tecnologiche, un approccio intuitivo potrebbe semplicemente riscalare la geometria e i parametri di tensione da una tecnologia esistente [25]. Per esempio, si potrebbe restringere la lunghezza effettiva del gate ($L_{eff}$), l'equivalente elettrico dello spessore dell'ossido ($T_{ox}$), la soglia di tensione($V_{th0}$), resistenza di canale e sorgente parassita($R_{dsw}$), e tensione fornita ai valori obiettivo($V_{dd}$) mentre gli altri valori vengono lasciati invariati [6].

Sebbene ci siano circa un centinaio di parametri in un modello BSIM per calcolare le caratteristiche I-V e C-V di un nanoscale transistor, solo una decina di essi sono critici per determinare il comportamento principale di un MOSFET durante il riscalamento tecnologico. $L_{eff}$,$T_{ox}$,$V_{th0}$,$R_{dsw}$ e $V_{dd}$ rappresentano il gruppo di parametri relativo alle specifiche di processo. Quando il loro valore viene determinato, durante il riscalamento della tecnologia, viene caratterizzata la specifica di base di una nuova tecnologia [25].

Per modellare l'incertezza dei parametri tecnologici dei dispositivi CMOS di un circuito è stata utilizzata la teoria degli insiemi fuzzy. Sfortunatamente, la progettazione con incertezza è computazionalmente intensiva e tipicamente richiede un costo computazionale maggiore rispetto al design deterministico. Per ridurre tale alto costo computazionale sono state utilizzate delle approssimazioni lineari.

Mantenendo costanti i parametri di design, la linearizzazione viene effettuata sui parametri di processo ed operativi. Mediante i coefficienti ottenuti con l'operazione di linearizzazione, viene effettuata la fuzzificazione delle risposte del circuito con il metodo dell'intervallo mediano [22].

Infine un test di riscalamento tecnologico è stato effettuato calcolando le misure di possibilità di fallimento rispetto a determinate specifiche e queste sono state minimizzate mediante il metodo numerico di ottimizzazione non lineare del simplesso di Nelder e Mead con Annealing

Simulato.

## 6.2 Descrizione del problema di ottimizzazione

Il processo di sizing dei dispositivi MOS della rete di circuito rappresentante l'OTA a due stadi mediante la metodologia di PWCD prende in considerazione i parametri da ottimizzare e i loro range sono mostrati nella tabella 6.1. I parametri indicati con "W" si riferiscono alla larghezza di canale del MOS, "R" (resistenza) e "C" (capacità) della rete esterna.

Tabella 6.1: Parametri da ottimizzare per l'applicazione OTA.

| **Parametri** | **Ranges** | **Unità** |
|---|---|---|
| **W1b=W1a** | 0.6-20 | $\mu m$ |
| **W3** | 0.6-20 | $\mu m$ |
| **W5** | 0.6-20 | $\mu m$ |
| **W4** | 0.6-20 | $\mu m$ |
| **W2b=W2a** | 0.6-20 | $\mu m$ |
| **C** | 1-15 | $pF$ |
| **R** | 2-40 | $K\Omega$ |

Le metriche di performance nella progettazione di amplificatori analogici per le ottimizzazioni effettuate mediante la metodologia di Possibilistic Worst–Case Distance, sono state le seguenti:

**Guadagno alle basse frequenze** si tratta del guadagno a 100Hz che è la base per un range di amplificazione.

**Margine di fase** il margine di fase è una misura di qualità per il circuito perché è relativo agli effetti parassiti come l'accoppiamento incrociato che causa il fallimento nel raggiungimento delle performance richieste.

**Frequenza all'unità di guadagno** definita come il valore di frequenza quando l'amplificatore ha almeno guadagno unitario.

**Consumo di potenza** il consumo di potenza è oggi molto importante per quanto riguarda il gran numero di applicazioni che funzionano a batteria.

Tale problema di ottimizzazione è stato formulato come problema a singolo obiettivo(SOP) in cui la funzione obiettivo da minimizzare è la somma delle possibilità di fallimento relative alle metriche di performance descritte sopra:

$$\sum_{i \in I} \Pi_{(P_i - P_{fi})}([0, +\infty]) \tag{6.1}$$

in tale funzione $I$ si riferisce all'insieme delle metriche di performance di circuito, $P_i$ è la performance i–esima di circuito e $P_{fi}$ è il fallimento i–esimo.

La somma delle possibilità è la norma $L_1$ nello spazio delle possibilità degli obiettivi del problema. La scelta di tale norma ha permesso la caratterizzazione delle regioni convesse del problema con più obiettivi in modo tale da avere un comportamento analogo al caso di un problema con singolo obiettivo.

I valori di fallimento $P_{fi}$ relativi alle performance di circuito sono riportati in tabella 6.2.

Tabella 6.2: Fallimenti relativi alle performance di circuito

| **Performance** $P_i$ | **Fallimento** $P_{fi}$ | **Unità** |
|---|---|---|
| Frequenza all'unità di guadagno | $> \quad 20$ | $MHz$ |
| Guadagno a 100 Hz | $> \quad 60$ | $dB$ |
| Margine di fase | $> \quad 60$ | $Degree$ |
| Consumo di Potenza | $< \quad 67$ | $mW$ |

## 6.3 Algoritmo di ottimizzazione utilizzato

L'algoritmo di ottimizzazione utilizzato nella minimizzazione della funzione 6.1 è una variante del metodo del simplesso di Nelder e Mead con Simulated Annealing.

Il *metodo del simplesso di Nelder e Mead* [18] è un metodo numerico di ottimizzazione non lineare. Tale metodo viene utilizzato per la minimizzazione di una funzione obiettivo in uno spazio $N$-dimensionale. Il metodo utilizza il concetto di simplesso che è un politopo di $N + 1$ vertici. Ad ogni iterazione si identifica il vertice peggiore (avente valore più alto della

funzione da minimizzare) e lo si riflette rispetto al centroide dei restanti $N$ vertici. Si identifica così un nuovo simplesso rispetto al quale proseguire la ricerca. Il simplesso può inoltre subire espansioni, contrazioni o riduzioni a seconda della situazione contingente.

La variante con *Simulated Annealing* [20], per analogia con il processo fisico di annealing (ricottura, termine usato in metallurgia), ad ogni passo dell'algoritmo rimpiazza la soluzione corrente con una soluzione random vicina, scelta con una probabilità che dipende da una differenza tra il corrispondente valore della funzione e da un parametro globale $T$ (chiamato temperatura). $T$ è stato posto pari a 4, cioè al valore massimo della funzione 6.1 da minimizzare. Il valore della temperatura viene fatto diminuire in maniera graduale dell'80% del suo valore ogni 15 iterazioni del metodo del simplesso. Il risultato che si ottiene è che la soluzione corrente cambia quasi casualmente quando $T$ è grande, ma tende sempre più a passi in 'discesa' al tendere di $T$ verso $0$. L'autorizzazione verso passi in 'salita' preserva il metodo dal rimanere bloccato in minimi locali.

## 6.4 Approssimazione lineare e fuzzificazione

Per modellare l'incertezza dei parametri di circuito operativi e di processo è stata utilizzata la teoria degli insiemi fuzzy. Poichè la progettazione per l'incertezza è computazionalmente intensiva e tipicamente richiede un costo computazionale maggiore rispetto al design deterministico sono state utilizzate delle approssimazioni. Per approssimare la risposta del circuito, rispetto ai parametri tecnologici ed ai parametri operativi, è stato utilizzata un'approssimazione lineare. Tale linearizzazione viene ottenuta attraverso un fitting su di un numero ristretto di simulazioni campione che vengono calcolate mantenendo fissi i parametri di design del circuito e facendo variare i parametri operativi e di processo mediante la procedura di campionamento dell' iper-cubo latino.

Formalmente, siano $\mathbf{w}$ i parametri di design: larghezza del canale dei mosfet, resistenza di compensazione e capacità di compensazione (vedi tabella 6.1); siano $\mathbf{o}$ i parametri operativi: temperatura $temp$ e tensione $V\_V1$; infine, siano $\mathbf{t}$ i parametri di processo o tecnologici: la lunghezza effettiva del gate $L_eff$, l'equivalente elettrico dello spessore dell'ossido $T_{ox}$, la soglia di tensione $V_{th0}$ e la resistenza source/drain per unità di lunghezza $R_{dsw}$. In tabella 6.3 vengono mostrati i ranges di variazione dei parametri operativi e di processo appena descritti.

Tabella 6.3: Valori dei parametri operativi e parametri di processo dell'OTA.

| **Parametri** | **Ranges** | **Unità** |
|---|---|---|
| $temp$ | 0 - 50 | $Degree$ |
| $V\_V1$ | 3.5 - 4.2 | $V$ |
| $L_eff$ | $0.9 \pm perc$ | $\mu m$ |
| $T_{ox}$ per NMOS | $9. \pm perc$ | $nm$ |
| $V_{th0}$ per NMOS | $0.6322 \pm perc$ | $V$ |
| $R_{dsw}$ per NMOS | $650 \pm perc$ | $\Omega \cdot \mu m$ |
| $T_{ox}$ per PMOS | $9. \pm perc$ | $nm$ |
| $V_{th0}$ per PMOS | $-0.6733 \pm perc$ | $V$ |
| $R_{dsw}$ per PMOS | $460 \pm perc$ | $\Omega \cdot \mu m$ |

L'incertezza dei parametri di processo è calcolata considerando una distribuzione statistica dei parametri di un certo valore in percentuale $perc$ rispetto al valore nominale di default della model card utilizzata[7]. Nei test effettuati sono state scelte le percentuali del $\pm 3\%$, $\pm 5\%$ e $\pm 7\%$ (si veda la tabella 6.4).

Le performance $\mathbf{P}$ del circuito: consumo di potenza, guadagno alle basse frequenze, frequenza all'unità di guadagno e margine di fase, si ricavano mediante la simulazione circuitale $\mathbf{Y}$.

$$\mathbf{P} = \mathbf{Y}(\mathbf{w}, \mathbf{o}, \mathbf{t}) \tag{6.2}$$

Mantenendo costanti i parametri di design, la linearizzazione viene effettuata sui parametri di processo ed operativi. In questo modo, la risposta viene approssimata attraverso un numero ristretto di simulazioni che vengono calcolate mantenendo fissi i parametri di design del circuito e facendo variare i parametri operativi e di processo mediante la procedura di campionamento dell' iper-cubo latino.

Il metodo statistico di *campionamento dell'iper-cubo* latino viene solitamente utilizzato per generare una distribuzione multi-dimensionale di una collezione di parametri. In un iper-cubo latino ciascun campionamento è l'unico nell'iperplano allineato con gli assi che lo contiene.

Formalmente la linearizzazione avviene mediante un operatore funzionale $Linearize$ come

segue:

$$\mathbf{a} = Linearize(\mathbf{w}, Y, \mathbf{I_o}, \mathbf{I_t}) \tag{6.3}$$

Dove $\mathbf{I_o}$ ed $\mathbf{I_t}$ sono gli intervalli di definizione dei parametri operativi e tecnologici (vedi tabella 6.3). Siano $\mathbf{a}$ i coefficienti risultanti dalla linearizzazione descritta sopra. Le risposte del circuito vengono rappresentate con il vettore di numeri aleatori $\hat{\mathbf{P}}_\mathbf{w}$ e sono calcolate come segue:

$$\hat{\mathbf{P}}_\mathbf{w} = a_0 + \sum_{i=1}^{n} a_i \mathbf{O}_i + \sum_{i=1}^{m} a_{n+i} \mathbf{T}_i; \tag{6.4}$$

dove *n* si riferisce al numero dei parametri operativi rappresentati dal vettore di numeri aleatori $\mathbf{O}_i$ mentre *m* è il numero dei parametri di processo rappresentati dal vettore di numeri aleatori $\mathbf{T}_i$.

I numeri fuzzy $\tilde{\mathbf{P}}$ associati alle performance di circuito vengono costruiti mediante il metodo dell'intervallo mediano [22], a partire da un insieme di realizzazioni dei vettori di numeri aleatori $\mathbf{O}_i$ e $\mathbf{T}_i$ attraverso un generatore di numeri casuali uniformemente distribuiti nei rispettivi intervalli di definizione (vedi tabella 6.3).

Il *metodo dell'intervallo mediano* consiste nel costruire il numero fuzzy mantenendo la proprietà dell'inviluppo convesso. In questo caso i numeri fuzzy sono definiti come mappe che associano ad ogni livello $\alpha \in [0, 1]$ un intervallo reale $A_\alpha$ tale che:

$$\alpha'' > \alpha' \Rightarrow A_{\alpha''} \subseteq A_{\alpha'} \tag{6.5}$$

Dato il numero fuzzy in forma di mappa ($\alpha$-livello, intervallo) si associa a ciascun $\alpha$-livello il più piccolo intervallo mediano che contiene una frazione di $(1 - \alpha)$ del set di dati $\hat{\mathbf{P}}_\mathbf{w}$.

## 6.5 Possibilità e ottimizzazione

Durante la procedura di riscalamento tecnologico dell'OTA a due stadi è stata tenuta in conto la misura di possibilità di fallimento rispetto a certe specifiche. Per calcolare la possibilità di fallimento si deve confrontare un numero crisp con un numero fuzzy. Si noti che un numero fuzzy può anche essere considerato come la traccia di una misura di possibilità sui singoli elementi $x$ di un insieme universale $X$ [11]. Quando viene considerata una misura di possibilità

definita sull'unità di intervallo, la sua distribuzione di possibilità $\pi$ può essere quindi interpretata come funzione di appartenenza di un numero fuzzy **B** descrivendo l'evento sul quale la misura di possibilità $\Pi$ si focalizza, come segue:

$$\Pi(\{x\}) = \pi(x) = \mathbf{B}(x), \ \ \forall x \in X. \tag{6.6}$$

La misura di possibilità che un numero crisp sia più piccolo o uguale ad un numero fuzzy **B** è definita da [11] come segue:

$$\Pi_{\mathbf{B}}([x, +\infty]) = \sup_{y \geq x} \mathbf{B}(y), \ \ \forall x. \tag{6.7}$$

La funzione di distribuzione di possibilità $\pi_{\mathbf{B}}$ corrispondente alla misura di possibilità della 6.7 è mostrato graficamente in figura 6.1 per il caso generale dove $\mathbf{B}$ ha una funzione di appartenenza non lineare.

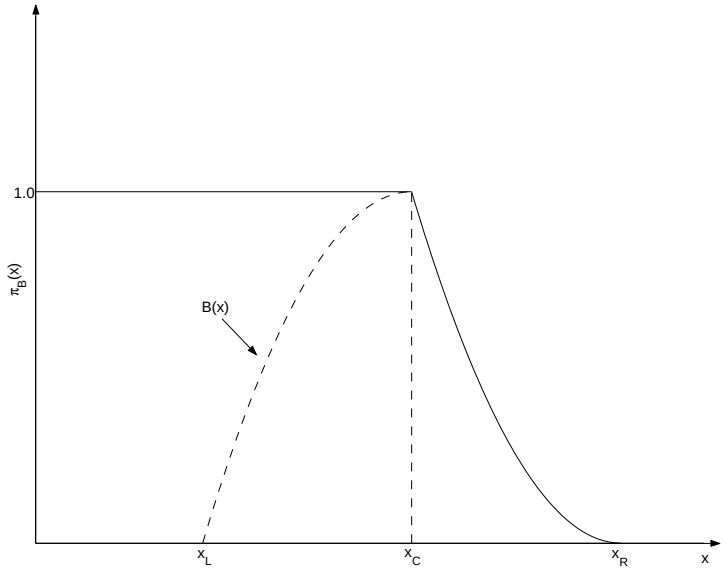



Figura 6.1: Distribuzione di possibilità di $\mathbf{B} \geq x$ per la funzione di appartenenza non lineare $\mathbf{B}(x)$, che ha supporto in $(x_L, x_R)$

Basandosi sulla 6.6 e su 6.7, la distribuzione di possibilità di fallimento $\pi_{(P-P_f)}$ è ottenuta dalla funzione fuzzy $(P - P_f)$ che contiene i numeri fuzzy $P$ (le performance di circuito) e $P_f$ (i fallimenti) come variabili. La possibilità di fallimento $(P - P_f \geq 0)$ è quindi definita come:

$$\Pi_{(P-P_f)}([0, +\infty]) = \sup_{y \geq 0}(P - P_f)(y) \tag{6.8}$$

## 6.6 Pseudocodice della metodologia PWCD

In questa sezione viene presentato lo pseudo codice della metodologia di Possibilistic Worst-Case Distance:

**Algorithm 4** Possibilistic Worst-Case Distance Optimization pseudocode

1: initialize simplex with $\mathbf{w}$
2: Input $T$, $iiter$ for SA
3: Run Simplex Method with SA
4: Print final simplex

$FuzzyCircuitFun$ è l'algoritmo che calcola la funzione da minimizzare mediante il metodo del simplesso con Simulated Annealing.

**Algorithm 5** $FuzzyCircuitFun$ ( $\mathbf{w}$ )

1: $\mathbf{inf}$,$\mathbf{sup}$ := ranges for $\mathbf{o}$ and $\mathbf{p}$
2: coefficients := $LinearizeFun$ ($\mathbf{w}$, $\mathbf{inf}$,$\mathbf{sup}$, $CircuitFun$)
3: r:= random variables between $\mathbf{inf}$ and $\mathbf{sup}$ uniformly generated
4: $\hat{\mathbf{P}}$:=$FuzzyfyFun$(coefficients, r)
5: Calculate the measures of possibility of failure given $\hat{P}$ and $P_{fi}$
6: Get the sum of the measures of possibilities

$LinearizeFun$ è la funzione che effettua una linearizzazione delle risposte del circuito facendo variare i parametri di processo e di design (si veda l'equazione 6.3).

**Algorithm 6** $LinearizeFun$ ($\mathbf{w}$, $\mathbf{inf}$,$\mathbf{sup}$, $CircuitFun$)

1: dim:=$m + n$
2: len:=$(m + n) * 2$
3: X:=$LatinizeFun$(dim, len, inf, sup)
4: P:= $CircuitFun$($\mathbf{w}$,parameters)
5: $\mathbf{a} := X \backslash P$ {'\' è l'operatore di divisione tra matrici del Matlab}
6: Return $\mathbf{a}$.

$LatinizeFun$ Calcola l'iper-cubo latino di dimensione $dim * len$.

$FuzzyfyFun$ è la funzione che restituisce le performance di circuito fuzzificate $\hat{P}$ mediante 6.4.

$CircuitFun$ è la funzione che, dati i parametri di design, di processo ed operativi del circuito, calcola le performance di circuito $P$ (vedi 6.2).

## 6.7 Risultati

La tabella 6.4 confronta due metodologie di design volte ad individuare design robusti rispetto ai modelli tecnologici e alle condizioni operative.

La colonna "Nominal Over-Design" presenta i risultati di yield della metodologia di progettazione più diffusa. Per il Nominal Over-Design ogni obiettivo viene fissato a un valore di sicurezza rispetto alla specifica nominale. In questo caso gli obiettivi sono stati aumentati del 10% rispetto alle soglie minime e diminuiti del 10% nel caso di soglie massime.

La colonna "PWCD" riporta i risultati del calcolo dello yield relativo alla metodologia di Possibilistic Worst-Case Distance.

Per entrambe le metodologie l'algoritmo di ottimizzazione utilizzato a supporto è la combinazione del metodo del simplesso di Nelder e Mead con Simulated Annealing. Lo scheduling della temperatura di annealing permette di coordinare la convergenza delle variabili di ottimizzazione rendendo globale la procedura di ricerca.

Il valore di yield è stato calcolato come rapporto tra il numero di chip che hanno tutte le performance accettabili sul totale di quelli prodotti (vedi relazione 4.5). Tale calcolo è stato effettuato mediante simulazione Montecarlo. Sono state effettuate 3 prove indipendenti considerando una distribuzione statistica dei parametri di processo del 3%,5% e 7% rispetto al valore nominale di default della model card utilizzata[7]. Sono state effettuate in totale 2000 simulazioni per ciascuna prova.

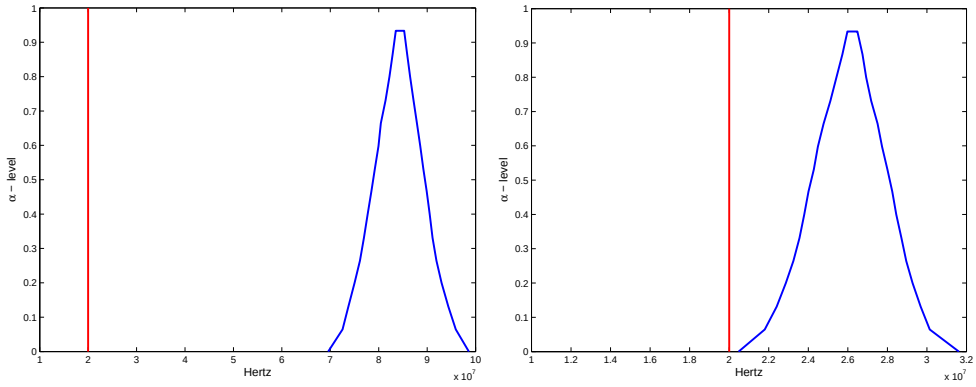
| **YIELD** | **Nominal over-design** | | | **PWCD** | | |
|---|---|---|---|---|---|---|
| | $1^a$ prova | $2^a$ prova | $3^a$ prova | $1^a$ prova | $2^a$ prova | $3^a$ prova |
| Variazione $\pm 3\%$ | 25.6% | 57.0% | 99.2% | 98.4% | 98.8% | 99.1% |
| Variazione $\pm 5\%$ | 88.4% | 64.5% | 49.1% | 98.7% | 98.7% | 98.1% |
| Variazione $\pm 7\%$ | 56.7% | 73.9% | 88.6% | 98.7% | 99.6% | 99.8% |

Tabella 6.4: Risultati Yield relativi ai due metodi Nominal over-design e Possibilistic Worst-Case Distance ottenuti mediante processo di simulazione Montecarlo.

La parte sinistra della tabella mostra come la metodologia di Nominal Over-Design porti ad una grande variabilità nei risultati di yield. Tale metodologia, ampiamente utilizzata, non dà quindi garanzie di robustezza nella progettazione. Invece, la colonna di destra, relativa alla metodologia di PWCD, riporta sempre design con alti valori di yield, quindi robusti rispetto alle incertezze prese in considerazione. Si noti inoltre che tali risultati sono stati ottenuti con un accettabile fattore di sovraccarico rispetto al numero di variabili di design, pari a due volte il numero di parametri di processo e operativi per ogni design da valutare.

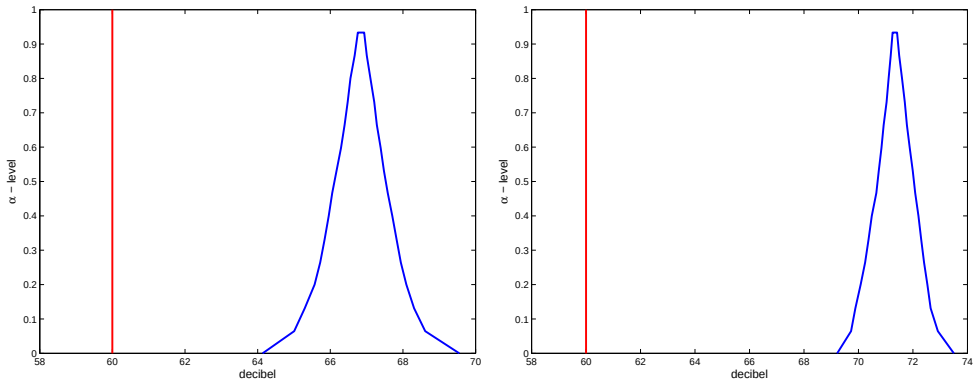
Nelle figure 6.2,6.3 ,6.4 e 6.5 sono rappresentati i numeri fuzzy relativi alle performance di circuito: Frequenza all'unità di guadagno, Guadagno a 100 Hz, Margine di fase e Consumo di Potenza rispetto ai valori di fallimento stabiliti in tabella 6.2. Le figure a sinistra $(a)$ rappresentano tali numeri fuzzy prima dell'ottimizzazione al passo di inizializzazione del simplesso, mentre le figure sulla destra $(b)$ rappresentano le performance circuitali ottenute alla fine dell'ottimizzazione.

Si noti anche che questo tipo di design elettronico ha tra gli obiettivi diversi trade-off che devono essere presi in considerazione dallo schema di ottimizzazione durante la ricerca del design ottimale. Data una configurazione iniziale in genere alcuni vincoli possono essere soddisfatti altri no. L'ottimizzazione deve quindi trovare una soluzione di soddisfacimento di tutte le specifiche contemporaneamente. La bontà della metodologia è mostrata dai grafici di destra che indicano che tutti i vincoli sono soddisfatti.



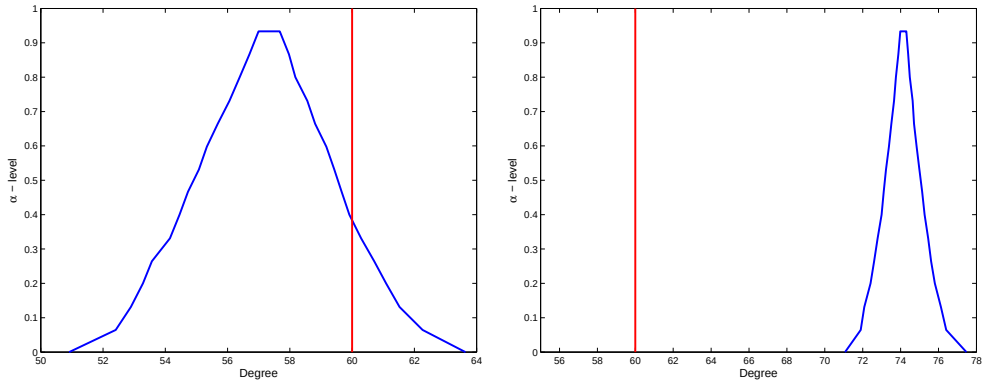

(a) (b)

Figura 6.2: Numero fuzzy relativo al Frequenza all'unità di guadagno (in blu) rispetto al valore di fallimento (in rosso) di $20MHz$ prima (a) e dopo (b) la procedura di ottimizzazione



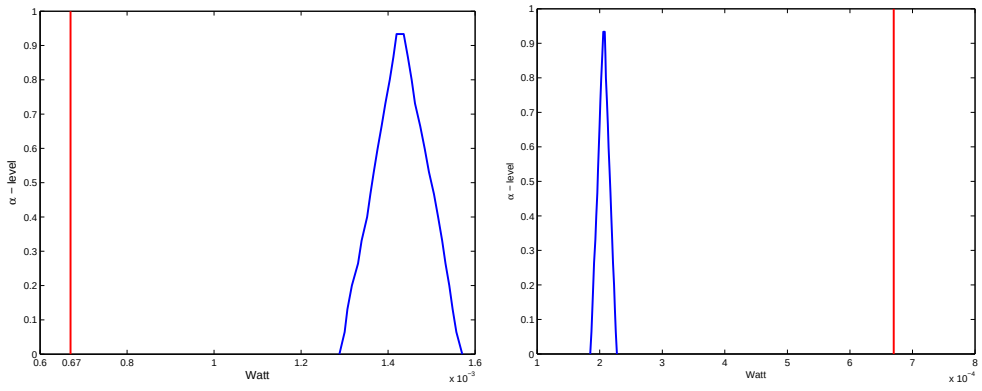

(a) (b)

Figura 6.3: Numero fuzzy relativo al Guadagno a 100 Hz (in blu) rispetto al valore di fallimento (in rosso) di $60dB$ prima (a) e dopo (b) la procedura di ottimizzazione




(a) (b)

Figura 6.4: Numero fuzzy relativo al Margine di fase (in blu) rispetto al valore di fallimento (in rosso) di $60Degree$ prima (a) e dopo (b) la procedura di ottimizzazione




(a) (b)

Figura 6.5: Numero fuzzy relativo al Consumo di Potenza (in blu) rispetto al valore di fallimento (in rosso) di $0.67mW$ prima (a) e dopo (b) la procedura di ottimizzazione

# Conclusioni

In questo lavoro di tesi sono state presentate due metodologie alternative a quelle più diffuse nell'industria microelettronica per il riscalamento dei circuiti analogici. L' applicazione sulla quale sono state testate tali metodologie è stata il sizing dei dispositivi MOS di un Operational Transconductance Amplifier a due stadi.

La prima metodologia analizzata è quella dell'utilizzo degli algoritmi evolutivi per l'ottimizzazione multiobiettivo. Il risultato di tale applicazione test è stato quello di ottenere una classe di dispositivi ottimizzati che coprono i range di performance ottimali rispetto ai trade-off di queste ultime.

Alla seconda metodologia proposta è stato dato il nome di Possibilistic Worst-Case Distance. Essa ha utilizzato i concetti di insieme fuzzy e teoria della possibilità per modellare l'incertezza dei parametri di circuito e per il calcolo e la minimizzazione della Worst-Case Distance. Tale metodologia ha mostrato i seguenti vantaggi :

- sono state modellate le incertezze che insorgono nel design circuitale con una metodologia che evita l'introduzione di concetti ed ipotesi statistiche;
- le caratteristiche di PWCD sono conformi a quelle richieste dalle metodologie integrate nei flussi di ottimizzazione, poichè la specifica del problema di worst-case distance è aderente a quella del design circuitale;
- la metodologia si è avvalsa solo dello schema di simulazione circuitale senza ulteriori strumenti di analisi dei circuiti analogici;
- la metodologia fornisce un approccio effettivo per ridurre il costo computazionale associato ad un design basato sull'utilizzo degli insiemi fuzzy per modellare l'incertezza.

# Bibliografia


[1] Ngspice. Web site of ngspice at http://ngspice.sourceforge.net/.

[2] H.L. Abdel-Malek and J.W. Bandler. Yield optimization for arbitrary statistical distributions: Part i - theory, part ii - implementation. *IEEE Transactions on Circuit and System*, CAS–27(4), 1980.

[3] A. M. Anile, V. Cutello, G. Nicosia, R. Rascuna', and S. Spinella. Comparison among evolutionary algorithms and classical optimization methods for circuit design problems. *Proceedings IEEE Press*, 1, 2005.

[4] K. Antreich, H. Graeb, M. Pronath, R. Schwencker, S. Zizala, and F. Schenkel. Mismatch analysis and direct yield optimization by specwise linearization and feasibility-guided search. *dac*, 00:858–863, 2001.

[5] K.J. Antreich, H.E. Graeb, and C.U. Wieser. Practical methods for worst-case and yield analysis of analog integrated circuits. *International Journal of High Speed Electronics and Systems*, 4(3):261–282, 1993.

[6] J.W. Bandler and S.H. Chen. Circuit optimization: The state of the art. *IEEE Transaction on Microwave Theory and Techniques*, 36(2), February 1988.

[7] Y. Cao, T. Sato, D. Sylvester, M. Orshansky, and C. Hu. New paradigm of predictive mosfet and interconnect modeling for early circuit design. *Proc. of IEEE CICC*, 2000.

[8] Y. Cheng and C. Hu. *MOSFET modeling & BSIM3 user's guide*. Kluwer, Norwell, MA, USA, 1999.

[9] V. Cutello, G. Nicosia, R. Rascunà, and S. Spinella. Optimizing inductor circuit and two-stage operational transconductance amplifier by means evolutionary and classical algorithms. *International Journal of Computational Science and Engineering*, 2006.

[10] K. Deb. *Multi-Objective Optimization using Evolutionary Algorithms*. John Wiley & Sons, Inc., 2001.

[11] D. Dubois and H. Prade. Possibility theory: An approach to computerized processing of uncertainty. *New York: Plenum Press*, 1988.

[12] D. Dubois and H. Prade. Probability theory and multiple-valued logics: A clarification. *Annals of Mathematics and Artificial Intelligence :, .*, 32(2–4):35–66, August 2001.

[13] H.E. Graeb, C.U. Wieser, and K.J. Antreich. Improved methods for worst-case analysis and optimization incorporating operating tolerances. pages 142–147, 1993.

[14] T. Hanne. Global multiobjective optimization using evolutionary algorithms. *Journal of Heuristics*, 6(3):347–360, 2000.

[15] E. Hjamlmarson. *Studies on desing automation of analog circuits – the design flow*. PhD thesis, Department of Electrical Engineering. Linlköpings universitet, Linlköpings, Sweden, 2003.

[16] G. Klir and B. Yuan. *Fuzzy sets and fuzzy logic: theory and applications*. Prentice-Hall, 1995.

[17] F. Medeiro-Hidalgo, R. Dominguez-Castro, A. Rodríguez-Vázquez, and J. L. Huertas. A prototype tool for optimum analog sizing using simulated annealing. *Proc. IEEE Int. Symp. Circuits Syst.*, pages 1933 – 1936, May 1992.

[18] J.A. Nelder and R. Mead. A simplex method for function minimization. *Computer Journal*, 7:308–313, 1965.

[19] Z.H. Liu C. Hu P.K. Ko, J.H. Huang. Bsim3 for analog and digital circuit simulation. In *IEEE Symp. on VLSI Technology CAD*, pages 400–429, January 1993.

[20] W.H. Press, S.A. Teukolsky, W.T. Vetterling, and B.P. Flannery. *Numerical Recipes in C++: The Art of Scientific Computing*. Cambridge University Press, 2002.

[21] Y. Sawaragi, H. Nakayama, and T. Tanino. *Theory of multiobjective optimization*. Academic Press Orlando, 1985.

[22] S. Spinella and A.M. Anile. Modeling uncertain sparse data with fuzzy b-splines. *Reliable Computing*, 10(5):335 – 355, October 2004.

[23] G. Venter and R.T. Haftka. Using response surface approximations in fuzzy set based design optimization. *Structural and Multidisciplinary Optimization*, 18(4):218–227, December 1999.

[24] L.A. Zadeh. Fuzzy sets as the basis for a theory of possibility. *Fuzzy Sets and Systems*, 1:3–28, 1978.

[25] W. Zhao and Y. Cao. New generation of predictive technology model for sub-45nm design exploration. *isqed*, 0:585–590, 2006.

[26] E. Zitzler. *Evolutionary Algorithms for Multiobjective Optimization: Methods and Applications*. PhD thesis, Swiss Federal Institute of Technology Zurych, Computing Engineering and Network Laboratory, Shaker Verlag,Aachen, Germany, December 1999.

# Ringraziamenti

A conclusione di questo lavoro di tesi sento il dovere di ringraziare tutti coloro che mi hanno fornito supporto tecnico (e non solo), durante il periodo di stesura della tesi e, in generale, durante il periodo di studi universitario.

Ringrazio il relatore della tesi, il professor Angelo Marcello Anile, per gli spunti scientifici fornitimi, di grande interesse culturale e personale.

Ringrazio il tutor del progetto, il dottor Salvatore Spinella, per gli stimoli a colmare le lacune riscontrate durante la stesura della tesi e per la chiarezza espositiva e la pazienza che mi ha accordato.

Ringrazio tutti i miei colleghi, dottoresse e dottori, in particolare Annalisa Cappello, Rosetta Rizzo, Marco Lucio Lovetere ed Enzo Tinè, per l'amicizia e per il continuo scambiarsi reciproco di conoscenze e passioni comuni.

Ringrazio la mia famiglia, i miei genitori e mio fratello per avermi sempre supportato e rassicurato.

Grazie Salvo, per essere riuscito a mantenere la calma anche quando le situazioni si facevano tese e difficili, per essere riuscito a trasformare ai miei occhi massicce muraglie in fragili steccati.