Università degli Studi di Trieste

♢

Facoltà di Ingegneria
Corso di Laurea in Ingegneria Elettronica

Tesi di Laurea
in
Teoria dell'Informazione e Codici

Minimi locali nelle reti di Hopfield

Laureando:
Daniele Casagrande

Relatore:
Chiar.mo Prof. Giuseppe Longo

Correlatore:
Prof. Francesco Fabris

---

anno accademico 1997/98

alla mia famiglia

Indice

Introduzione

Le reti neurali artificiali sono sistemi costruiti sulla base degli stessi principi che si ipotizza stiano a fondamento del comportamento del cervello umano. Esse costituiscono un oggetto di indagine che si colloca tra la neurologia, che studia la struttura e la morfologia dei neuroni naturali, la psicologia, che si occupa delle caratteristiche macroscopiche delle reti di neuroni naturali e l'intelligenza artificiale, che cerca di emulare i comportamenti intelligenti attraverso dispositivi meccanici ed elettrici.
Da quando, nel dopoguerra, cominciarono a comparire i primi manufatti neurali, l'interesse nei loro confronti ha conosciuto fasi alterne. A un periodo iniziale di entusiasmo fecero seguito forti critiche che misero in luce i limiti dei primi modelli e che causarono una battuta d'arresto di una ventina di anni nello sviluppo dei sistemi neuromorfici. In anni recenti, lo studio di questi modelli ha trovato nuovo vigore soprattutto grazie alle conquiste teoriche di Hopfield che, assieme all'idea dell'algoritmo di retropagazione, hanno ampliato il loro campo applicativo.
In questo lavoro viene studiato un modello particolare di reti neurali, proposto in origine da Hopfield per simulare la capacità mnemonica del cervello e poi utilizzato, in primo luogo dallo stesso Hopfield, per risolvere problemi di ottimizzazione combinatoria. Piú precisamente, si valuta la possibilità di costruire reti neurali capaci di risolvere in poco tempo - sarà chiarito in seguito, prendendo spunto della teoria della complessità computazionale, che cosa si intende per 'poco' - il problema del fusto, in inglese *knapsack problem*, sul quale si basa il cifrario omonimo.
Il primo capitolo contiene una breve ed essenziale panoramica sui principali modelli neurali esistenti, mentre la descrizione dettagliata del modello di Hopfield, nelle due versioni, discreta e continua, è affidata al capitolo successivo

nel quale si spiega come sia possibile costruire una rete che risolve il problema del fusto.
Per i modelli di Hopfield è possibile definire una funzione energetica, non crescente col tempo, associata all'evoluzione della rete. Nel caso in cui il modello sia utilizzato per realizzare una memoria associativa, vi è una corrispondenza tra i dati memorizzati e i minimi della funzione energia. Nel caso in cui, invece, esso sia costruito per risolvere un problema di ottimizzazione combinatoria, ogni minimo è associato a una risposta ma solo il minimo assoluto coincide con la soluzione ottima, mentre i minimi locali sono associati a risposte approssimate.
Il terzo capitolo della tesi verte su questi concetti e sui principali metodi oggi in uso per evitare che la rete finisca in un minimo locale. Nel quarto capitolo si propone e si studia un algoritmo nuovo basato su un'analisi geometrica della funzione energia e nel quinto si verificano le potenzialità di tale algoritmo applicandolo sia alla versione 'facile' del problema del fusto sia a quella 'difficile' - anche questi due aggettivi saranno chiariti in seguito - analizzandone i successi e giustificandone i limiti. In appendice, infine, sono raccolti i programmi utilizzati per simulare al calcolatore l'algoritmo e la rete neurale.

Capitolo 1

Le reti neurali artificiali

1.1 Generalità

Intorno agli anni quaranta furono gettate le basi per la definizione di un modello computazionale altamente parallelo che trae ispirazione dalle ipotesi sul funzionamento del cervello umano: le *reti neurali artificiali* (RNA) o piú semplicemente *reti neurali* (RN) [1]. Il motivo principale del rinnovato interesse degli studiosi è il fatto che tanto i sistemi dell'informatica tradizionale quanto quelli della piú nuova *intelligenza artificiale* (IA) sono incapaci di risolvere in maniera efficace alcuni tipi di problemi quali, ad esempio, il riconoscimento di immagini o la predizione dell'andamento di un segnale. Al contrario, il cervello umano è in grado di risolvere in maniera eccellente proprio questo tipo di problemi. Le RN nascono, allora, come alternativa sia alla computazione sequenziale classica, basata su algoritmi, che a quella dell'IA [2], basata su una rappresentazione simbolica della realtà.
Il concetto piú innovativo introdotto in questi nuovi modelli è il processo di apprendimento, grazie al quale essi sono in grado di acquisire un certo grado di conoscenza e di esperienza riguardanti il mondo esterno. Questa caratteristica deriva dalla struttura fisica della RN nella quale l'elaborazione dell'informazione avviene in maniera parallela coinvolgendo un elevato numero di connessioni tra unità elementari, i *neuroni*, che sono in grado di compiere solamente azioni molto semplici, come la ricezione di segnali dalle connessioni, l'esecuzione di piccoli calcoli booleani o aritmetici e l'invio di

segnali alle altre unità; in questo modo, ogni problema viene scomposto in problemi piú piccoli, risolti dai singoli neuroni. Inoltre i neuroni sono in grado di memorizzare una limitata quantità di informazioni, di solito il valore di una soglia di attivazione, mentre il resto delle informazioni necessarie alla rete per il suo funzionamento è memorizzato nel valore delle connessioni e nella loro topologia.
Un aspetto importante di questo paradigma computazionale è il carattere non locale della conoscenza del problema, che emerge come proprietà collettiva e macroscopica del sistema rendendolo quasi immune da malfunzionamenti che si possono verificare a livello microscopico e fornendolo di notevole tolleranza nei confronti di dati affetti da rumore. Inoltre le connessioni trasportano simultaneamente i loro segnali cosí come i neuroni possono utilizzarli contemporaneamente, e ciò assicura un'elevata velocità di risposta.
Le operazioni che una RN può eseguire sono le piú diverse e si possono riassumere nei seguenti punti:

- *classificazione:* si utilizza una quantità rappresentativa di ingressi per costruire, un certo numero di classi di appartenenza e agli ingressi successivi la rete risponde con la classe corrispondente
- *accoppiamento ingresso-uscita:* la rete memorizza un certo numero di coppie ingresso-uscita ed a un particolare ingresso risponde con l'uscita ad esso accoppiata
- *completamento:* presentando all'ingresso un dato incompleto, la rete ne ricostruisce la parte mancante sfruttando una memoria sviluppata precedentemente
- *ottimizzazione:* si fornisce alla rete un ingresso che rappresenta i valori iniziali di un particolare problema di ottimizzazione combinatoria e la rete produce un insieme di uscite che codifica la soluzione
- *controllo:* l'ingresso rappresenta lo stato corrente di un controllore e l'uscita corrisponde alla sequenza di comando desiderata.

A fronte di tanti vantaggi, vi è lo svantaggio della notevole complessità di un modello cosí strutturato e della conseguente impossibilità di comprenderne il funzionamento globale; si può soltanto dimostrarne la verosimiglianza

partendo da ipotesi sul comportamento delle singole unità e sulla topologia delle connessioni.

1.2 L'unità elementare: il neurone

In una RN biologica, ogni neurone è composto da un corpo cellulare detto *soma*, che scambia informazioni con le altre unità elementari, ricevendole tramite i *dendriti* e spedendole, dopo averle elaborate, lungo l'*assone* (vedi Figura 1.1). Il collegamento fra due neuroni avviene in una zona che prende

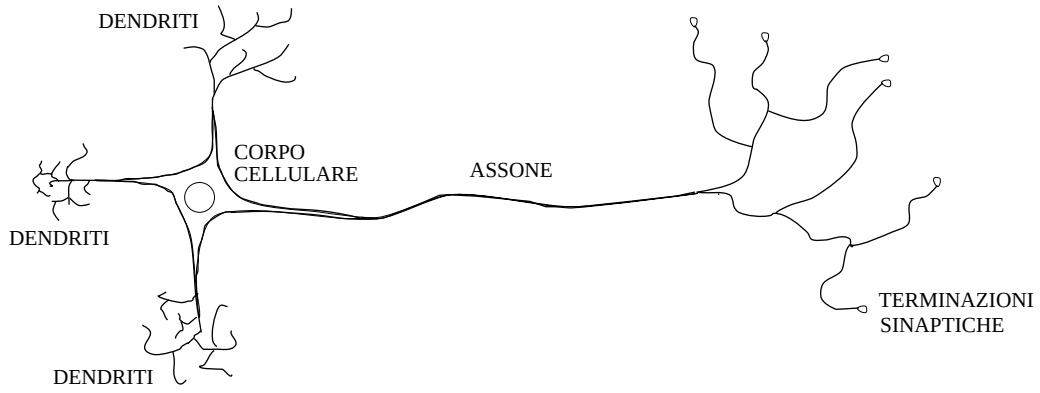


Figura 1.1: Rappresentazione di un neurone naturale.

il nome di *area sinaptica* (o *sinapsi*) nella quale l'assone del neurone a monte si collega con un dendrite del neurone a valle; il contatto può essere *eccitatorio* o *inibitorio* a seconda che contribuisca positivamente o negativamente all'attività del secondo neurone. Inoltre, più l'area sinaptica è estesa, maggiore è la dipendenza dello stato del neurone a valle dall'uscita del neurone a monte; nel corpo cellulare, infatti, viene effettuata la somma di tutti i segnali provenienti dai dendriti, ottenendo cosí un valore di *attivazione* dal quale dipende in modo non lineare l'uscita del neurone, cio il livello del segnale da inviare lungo l'assone. La codifica dell'informazione avviene attraverso una modulazione di posizione di un treno di impulsi generato dal neurone e inviato lungo l'assone [3].

Un sistema matematico che simuli il comportamento di un neurone naturale, dovrà realizzare ciascuna delle funzioni elementari appena descritte; uno dei primi e piú famosi modelli è quello rappresentato in Figura 1.2. Il valore di

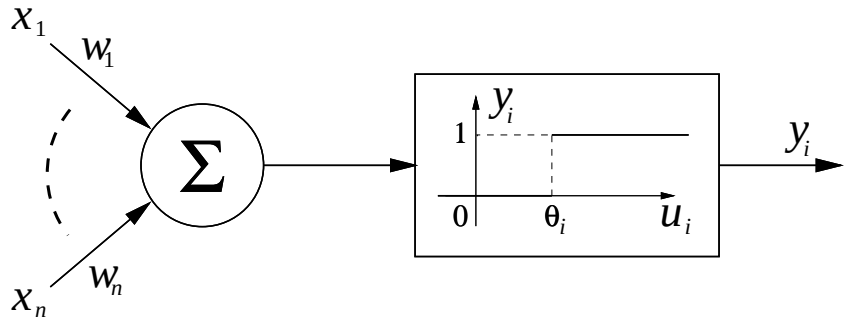


Figura 1.2: Il modello neurale di McCulloch e Pitts.

attivazione u_i del neurone i-esimo è dato dalla somma algebrica dei segnali provenienti dagli altri neuroni x_j e pesati dalla forza sinaptica w_j; poiché w_j assume valori maggiori o minori di zero, a seconda che il contatto sia eccitatorio o inibitorio, u_i può risultare sia positivo che negativo. L'uscita del neurone è 0 (neurone *inattivo*) o 1 (neurone *attivo*) a seconda che il livello di attivazione sia maggiore o minore di un valore di *soglia* θ_i che a sua volta può essere sia positivo che negativo:

$$y_i = \begin{cases} 0 & \text{se } \sum_j w_j x_j \leqslant \theta_i \\ 1 & \text{se } \sum_j w_j x_j > \theta_i \end{cases} \tag{1.1}$$

In un articolo del 1943 [4], Warren McCulloch e Walter Pitts, basandosi su questo modello di attività nervosa (detta del *tutto o niente*), studiarono gli eventi neurali e le relazioni tra essi con i metodi della logica proposizionale e dimostrarono che una rete composta di neuroni di questo tipo (detti *binari* in quanto la loro uscita può assumere solo due valori) è in grado di eseguire qualunque computazione effettuata da una macchina di Turing.
Nei modelli successivi e piú evoluti, il blocco di uscita realizza una funzione di trasferimento non lineare che può assumere valori in tutto l'intervallo $[0, 1]$ presentando un andamento quasi lineare nell'intorno di θ_i e una saturazione per valori di u_i tendenti a $+\infty$ o $-\infty$; le funzioni piú utilizzate sono

- la *spezzata* (Figura 1.3(a)),

$$y_i = \begin{cases} 0 & \text{se } u_i \leqslant \theta_i - \alpha \\ \frac{1}{2\alpha}(u_i + \alpha - \theta_i) & \text{se } \theta_i - \alpha \leqslant u_i \leqslant \theta_i + \alpha \\ 1 & \text{se } \theta_i + \alpha \leqslant u_i \end{cases} , \tag{1.2}$$

- la *sigmoide* (Figura 1.3(b)),

$$y_i = \frac{1}{1 + e^{-\alpha(u_i - \theta_i)}} = \frac{1}{2}\left(\tanh\left[\frac{\alpha}{2}(u_i - \theta_i)\right] + 1 \right) , \tag{1.3}$$

- il *rapporto di funzioni quadratiche* (Figura 1.3(c)),

$$y_i = \begin{cases} \frac{(u_i - \theta_i)^2}{1 + (u_i - \theta_i)^2} & \text{se } u_i \geqslant \theta_i \\ 0 & \text{altrimenti.} \end{cases} \tag{1.4}$$

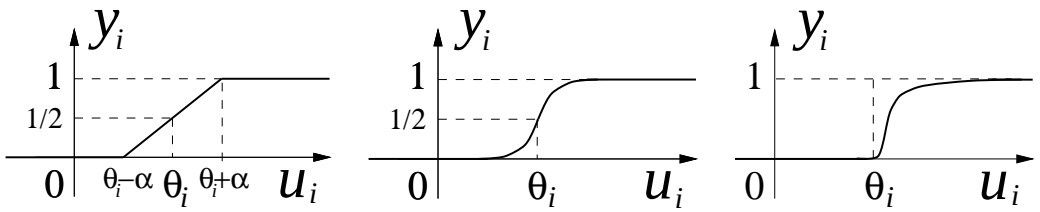


Figura 1.3: Le funzioni non lineari piú usate.

1.3 Modelli di reti neurali

Descritta l'unità base e chiariti alcuni concetti fondamentali, chiameremo *rete neurale* un insieme di neuroni collegati tra loro tramite connessioni monodirezionali. La connessione tra i neuroni i e j sarà in seguito caratterizzata da un'intensità (o forza) w_{ij} (in generale $w_{ij} \neq w_{ji}$).
Utilizzando i termini della teoria dei grafi, una RN si definisce come un grafo diretto pesato $G = (P, A, W)$ dove P è un insieme di nodi (i neuroni), A è un insieme di archi orientati (le connessioni) e W è un insieme di pesi, ognuno dei quali è associato a un arco.
In una rete vi possono essere neuroni che ricevono segnali dal mondo esterno

(detti *neuroni di ingresso*) e neuroni che vi inviano segnali (detti *neuroni di uscita*); i neuroni che non comunicano con il mondo esterno, vengono detti *nascosti*. La rete può essere *stratificata* se le unità elementari sono suddivise in *strati* distinti, tali cio che tutte le connessioni uscenti da ciascuna unità la collegano con unità dello stesso strato o dello strato successivo.
L'insieme ordinato di tutte le uscite dei neuroni rappresenta lo *stato* della rete ed è indicato con $\mathbf{V}$; il corrispondente spazio è lo *spazio degli stati* Ω. Analogamente si possono definire un sottospazio degli ingressi Ω_e di dimensione n_e e un sottospazio delle uscite Ω_u di dimensione n_u; l'*evoluzione* della rete corrisponde alla traiettoria percorsa da $\mathbf{V}$ in Ω: ad ogni istante, la posizione assunta dal vettore degli stati dipende dalla posizione all'istante precedente secondo una regola di evoluzione descritta, di solito, da una funzione di transizione a livello dei singoli neuroni. Quando la posizione di $\mathbf{V}$ non cambia nel tempo, si dice che la rete ha raggiunto uno stato di *equilibrio*.
A seconda della topologia delle connessioni, le reti si diranno:

reazionate o **ricorrenti** se esiste almeno un'unità la cui uscita è riportata in ingresso o se almeno due strati sono collegati in modo bidirezionale

non ricorrenti o **unidirezionali** se non esiste nessun tipo di ciclo ovvero se la trasmissione dell'informazione avviene solo in un senso: dall'ingresso verso l'uscita (in inglese questo tipo di reti è detto *feed forward*).

Come si è accennato nell'introduzione, una delle caratteristiche peculiari delle RN è la loro capacità di *apprendimento*; con questo termine si intende la modalità con cui, in un primo tempo, vengono ricavati i pesi w_{ij} delle connessioni affinché, durante l'impiego, la rete funzioni nel modo voluto; l'apprendimento può essere di due tipi:

supervisionato: viene individuato un insieme S di K coppie ingresso-uscita, $S = \left\{(\mathbf{x}_k, \mathbf{y}_k^d) : \mathbf{x}_k \in \Omega_e, \mathbf{y}_k^d \in \Omega_u, k = 1, \ldots, K\right\}$, denominato *insieme di apprendimento* (in inglese *training set*) e i valori w_{ij} sono aggiustati facendo coincidere l'uscita della rete con quella desiderata; terminata questa prima fase, ad un ingresso $\mathbf{x}_k$ la rete risponde con l'uscita $\mathbf{y}_k^d$.

non supervisionato: l'insieme di apprendimento è costituito da vettori del sottospazio Ω_e scelti in maniera rappresentativa; la rete classifica gli

ingressi in base alle correlazioni esistenti tra essi e durante l'impiego, presentando in ingresso un determinato $\mathbf{x} \in \Omega_e$, essa risponde con segnali diversi a seconda della classe a cui appartiene $\mathbf{x}$.

1.3.1 Il percettrone

Questo termine è la traduzione dell'inglese *perceptron* coniato negli anni cinquanta da Frank Rosenblatt [5] per indicare un modello di RN non reazionata, senza strati nascosti, con uno strato di ingresso costituito da n neuroni e un solo neurone di uscita (Figura 1.4).

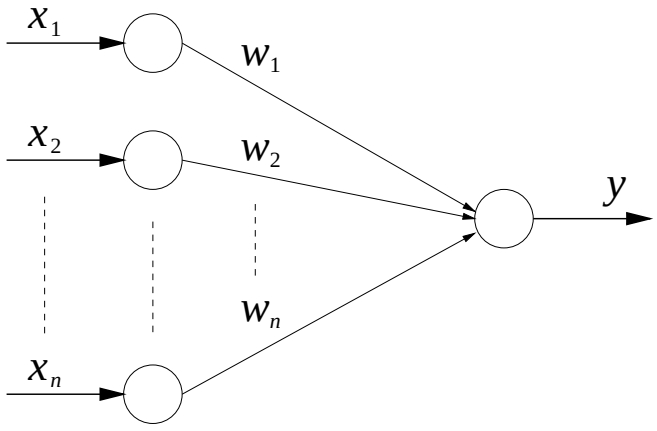


Figura 1.4: Architettura di un percettrone.

L'algoritmo di apprendimento di questo modello non ricorrente, detto *delta rule* o *regola delta*, si basa su un lavoro di Bernard Widrow e Marcian Hoff [6]: dato l'insieme S, si calcola l'errore come differenza tra l'uscita desiderata $y^d(t)$ all'istante t e l'uscita fornita dalla rete $y(t)$:

$$\Delta = y^d(t) - y(t) \quad . \tag{1.5}$$

La regola di aggiornamento dei pesi,

$$w_i(t+1) = w_i(t) + \eta \Delta x_i(t) \quad , \tag{1.6}$$

tiene conto di questa differenza mediante il parametro $\eta \in [0,1]$ che controlla la velocità di aggiornamento ed è detto *tasso* (o *fattore*) *di apprendimento*. La risposta della rete a un ingresso generico $\mathbf{x}$ è data da:

$$y = g\Big(\sum_{i=1}^{n} w_i x_i\Big) = g(\mathbf{W}\mathbf{x}) \tag{1.7}$$

dove $\mathbf{W} = (w_1, ..., w_n)$ è la matrice (in questo caso è un semplice vettore) delle connessioni e g è una delle funzioni non lineari di Figura 1.3.
La funzione (1.7) può essere rappresentata come un iperpiano che divide in due parti lo spazio Ω_i degli ingressi: ogni punto $\mathbf{x}$ appartiene all'uno o all'altro dei due semispazi a seconda che l'uscita corrispondente sia 0 oppure 1. Questa è la caratteristica piú importante del percettrone e viene detta *separazione lineare*; essa ne identifica i limiti in quanto il modello è in grado di risolvere solo i problemi linearmente separabili. A tal proposito, Tom Cover ha dimostrato che se esiste una soluzione al problema, l'algoritmo di apprendimento converge ad essa in un numero finito di passi (teorema di convergenza) [7].

1.3.2 Reti non reazionate multistrato

Questa limitazione del percettrone a un solo strato può essere superata utilizzando reti a piú strati, nelle quali gli strati nascosti servono per elaborare risposte parziali sulla base delle quali la rete fornirà la risposta finale. Un risultato interessante che dimostra le potenzialità di queste reti, anche se non fornisce indicazioni per determinare a priori il numero di strati e di neuroni nascosti necessari per realizzare la funzione desiderata, è il teorema di Kolmogorov, la cui dimostrazione si trova in [8]:

Teorema 1.1. *Data una qualunque funzione* $f : [0,1]^n \to \mathbb{R}^m$*, essa può essere realizzata esattamente da una RN non ricorrente a* 3 *strati con* n *unità nello strato di ingresso* $2n+1$ *in quello di elaborazione e* m *in quello di uscita.*

Per una RN di questo tipo si può definire una funzione di errore [9]:

$$e = \frac{1}{2}\sum_{i=1}^{n_u}(y_i^d - y_i)^2 \tag{1.8}$$

e l'apprendimento avviene tramite la propagazione di questo errore dall'uscita verso gli strati più interni e la conseguente modifica dei pesi delle connessioni; l'errore commesso dalla rete, infatti, dipende unicamente dalla matrice $\mathbf{W}$ e gli elementi w_{ij} saranno modificati in modo da rendere minimo tale errore. Su queste considerazioni si basa l'algoritmo proprosto nel 1986 da David Rumelhart, Geoffry Hinton e Ronald Williams [10] detto *back propagation* (in italiano *retropagazione*); il valore iniziale delle connessioni è scelto a caso e poi, per ogni coppia ingresso-uscita $(\mathbf{x}_k, \mathbf{y}_k^d)$, si eseguono le seguenti operazioni:

1. si fa evolvere il sistema ottenendo l'uscita $\mathbf{y}_k$
2. si calcolano i termini $\delta_i^k = (y_i^{dk} - y_i^k) y_i^k (1 - y_i^k)$ per $i = 1, ..., n_u$
3. si retropagano questi ultimi ricavando i termini

$$\delta_j^k = (\sum_{i=1}^{n_h} \delta_i^k w_{ji}) h_j^k (1 - h_j^k)$$

   per ciascuno dei nodi dello strato immediatamente precedente a quello di uscita ($\mathbf{h}$ è il vettore delle uscite di tale strato nascosto e n_h è la sua dimensione)
4. si ripete il passo 3 fino a raggiungere lo strato di ingresso
5. si aggiornano i pesi secondo la formula: $w_{ij}^k = w_{ij}^{k-1} + \eta \delta_j^k h_i^k$

L'importanza delle RN RP (cioè reti non reazionate che utilizzano l'algoritmo di retropagazione appena descritto) è sancita dal teorema di Hecht-Nielsen [9]:

Teorema 1.2. *Data una arbitraria funzione $f : [0,1]^n \to \mathbb{R}^m$ di tipo L_2 e un arbitrario $\epsilon > 0$, esiste una rete RP a tre strati che approssima la funzione f con errore quadratico medio minore di ϵ.*

La classe L_2 comprende, sostanzialmente, tutte le funzioni che possono essere riscontrate in problemi pratici come, per esempio, le funzioni continue e quelle con un numero finito di punti di discontinuità in $[0,1]$.

1.3.3 Reti autoassociative

Introdotte nei primi anni settanta da Teuvo Kohonen [11], queste reti sono costituite da due strati (vedi Figura 1.5): lo strato dei *recettori*, che ricevono gli ingressi dal mondo esterno, e lo strato degli *associatori*, che forniscono le uscite. Esse utilizzano un apprendimento non supervisionato basato su

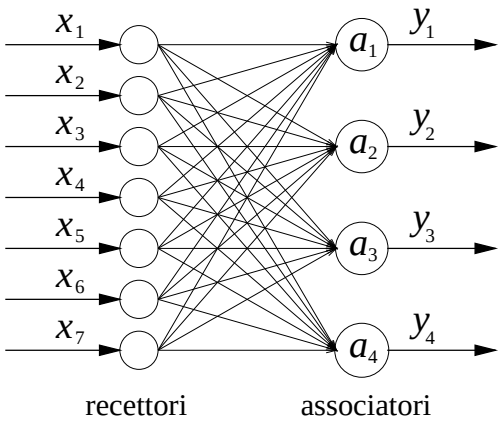


Figura 1.5: Una rete autoassociativa con 7 recettori e 4 associatori.

un algoritmo *competitivo* che provvede alla selezione di uno o piú associatori detti *vincitori* in base alle seguenti regole:

1. si sceglie a caso la matrice iniziale $\mathbf{W}$ dei pesi (w_{ij} è la connessione tra l'i-esimo recettore e il j-esimo associatore).

2. ad ogni ingresso $\mathbf{x}_k$ presentato alla rete, essa aggiorna la colonna $\mathbf{w}_j$ di $\mathbf{W}$, relativa al j-esimo associatore, secondo la formula:

$$\mathbf{w}_j^k = \alpha_j(d,k)\mathbf{x}_k + [1 - \alpha_j(d,k)]\mathbf{w}_j^{k-1}$$

   dove $\alpha_j(d,k)$ è il fattore di apprendimento.

Con il primo passo si suddivide lo spazio degli ingressi in regioni dette *insiemi di prossimità* ciascuna delle quali è associata a un nodo di uscita a_j nel senso che prendendo come ingressi due punti della stessa regione si ottengono valori di uscita che selezionano, in entrambi i casi, il nodo a_j (molto spesso

i valori di uscita degli associatori diversi da a_j sono molto piccoli, mentre a_j ha un'uscita prossima a uno).
All'inizio queste regioni non sono ben definite ma a mano a mano che si procede con le iterazioni della seconda regola, il sistema diventa sempre piú selettivo; il fattore $\alpha_j(d,k)$, infatti, decresce col passare del tempo e all'aumentare della distanza d tra un nuovo ingresso e quello memorizzato che diventerà un *attrattore* per la regione di prossimità ad esso associata.
Un modello particolare di apprendimento competitivo è la *corrispondenza spaziale* (in inglese *feature mapping*) che consente alla rete di dare informazioni relative alla struttura dei dati in ingresso: definendo per i neuroni di uscita un opportuno ordinamento spaziale, si può fare in modo che insiemi di prossimità vicini in Ω siano selezionati da associatori spazialmente vicini nello strato di uscita. In tal modo si introduce un isomorfismo tra la struttura dello strato di uscita e la geometria dello spazio degli stati, molto simile alla corrispondenza biologica tra stimoli sensoriali e aree cerebrali [1].

1.3.4 Memorie associative bidirezionali

Nel 1982 John Hopfield pubblicò un breve articolo [12] che ebbe grande risonanza tra gli addetti ai lavori perché presentava un modello di RN totalmente reazionato - tale, cioè, che per ogni coppia di nodi (p,q) esistono un arco che collega p a q, con peso w_{pq}, e un arco che collega q a p, con peso w_{qp} - il cui comportamento può essere studiato con i mezzi della teoria dei sistemi non lineari.
Egli mostrò che in certi casi l'evoluzione di una RN non lineare può essere descritta qualitativamente associando ad ogni stato raggiunto dal sistema un valore di energia che decresce col procedere dell'evoluzione e raggiunge un valore minimo in corrispondenza a uno stato stabile.
Ma il paradigma di Hopfield conteneva anche un'importante novità epistemologica. L'approccio tradizionale allo studio delle RN, come abbiamo visto, prevedeva l'applicazione di certe regole di apprendimento, di solito basata sulla modifica del valore delle connessioni, e una successiva analisi degli effetti di tale modifica. Hopfield, al contrario, sostenne che dato un insieme S di punti di Ω, è possibile costruire una RN per la quale essi risultino stati

stabili, nel senso che se si fa partire il sistema da un punto non appartenente a S, esso converge verso un punto di S.
Per ogni stato stabile $\mathbf{V}^s \in S$, si definisce il *dominio* (o *regione*) *di attrazione* (di $\mathbf{V}^s$) come l'insieme dei punti di Ω che, assegnati come stati di partenza della rete, la fanno evolvere verso $\mathbf{V}^s$. In realtà, la presenza di anelli chiusi implica, in generale, problemi di stabilità, ma introducendo opportune ipotesi la convergenza verso stati gli stabili può essere dimostrata rigorosamente. Rimandando al prossimo capitolo le ipotesi di lavoro da cui parte Hopfield, concludiamo questo paragrafo riportando un teorema dimostrato da Bart Kosko [13] valido per un particolare tipo di memorie associative dette, in inglese, *bidirectional associative memories* (BAM) (vedi Figura 1.6):

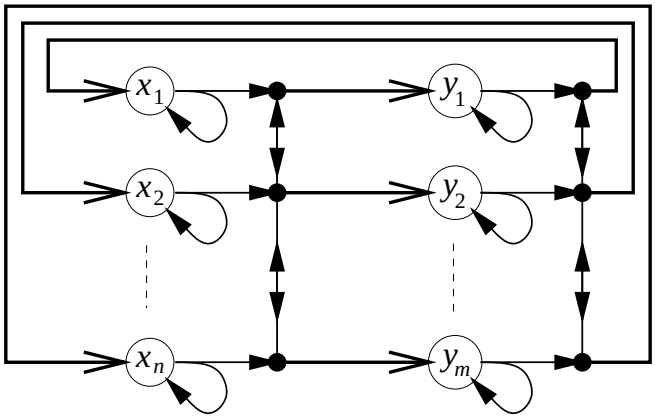


Figura 1.6: Struttura di una rete ricorrente di Kosko.

Teorema 1.3. *Una RN costituita da due insiemi di elementi di processo costituiti, rispettivamente, da n unità con uscite* $x_1, ..., x_n$ *e da m unità con uscite* $y_1, ..., y_m$ *e le cui funzioni di trasferimento siano:*

$$\dot{x_i} = -a_i x_i + \sum_{j=1}^{m} w_{ij} g(y_j) + S_i \qquad con \quad a_i, S_i > 0 \quad \forall i = 1, \dots, n$$

$$\dot{y_j} = -b_j y_j + \sum_{i=1}^{n} w_{ij} g(x_i) + T_j \qquad con \quad b_j, T_j > 0 \quad \forall j = 1, \dots, m$$

dove g è la solita funzione non lineare e $\boldsymbol{W}$ è una matrice $n \times m$, converge verso un punto stabile, indipendentemente dallo stato di partenza.

Capitolo 2

Le reti di Hopfield

2.1 Il modello discreto

La prima RN proposta da Hopfield è costituita da unità elementari di tipo binario come quelle studiate da McCulloch e Pitts: ogni neurone p_i ($i = 1, \ldots, n$), cioè, possiede solo due stati: $V_i = 0$ o $V_i = 1$; inoltre la regola di transizione, che descrive a livello microscopico l'evoluzione della rete, è la (1.1) che riscriviamo con le stesse notazioni utilizzate da Hopfield:

$$V_i = \begin{cases} 0 & \text{se } \sum_{j \neq i} T_{ij} V_j \leqslant U_i \\ 1 & \text{altrimenti.} \end{cases} \tag{2.1}$$

Secondo questa regola, ogni neurone cambia il proprio stato in modo asincrono, cioè in un istante casuale e senza relazione temporale con le altre unità elementari, in funzione del risultato del confronto tra la somma pesata degli ingressi e la soglia U_i e, ad ogni istante, allo stato del sistema corrisponde una n-upla binaria $\mathbf{V}$ i cui elementi rappresentano le uscite dei vari neuroni. Si supponga, ora, di voler memorizzare un insieme di stati $S = \left\{\mathbf{V}^s : \mathbf{V}^s \in \Omega, s = 1, \ldots, K\right\}$ di cardinalità K. Per farlo si può costruire la matrice dei pesi delle connessioni $\mathbf{T}$ secondo la seguente *regola di memorizzazione*:

$$T_{ij} = \begin{cases} \sum_{s=1}^{K} (2V_i^s - 1)(2V_j^s - 1) & \text{se } i \neq j \\ 0 & \text{se } i = j \end{cases} \tag{2.2}$$

da cui si ricava la soglia di attivazione per il neurone i-esimo quando la rete si trova nello stato $\mathbf{V}^{s'}$,

$$\sum_{j=1}^{n} T_{ij} V_j^{s'} = \sum_{s=1}^{K} (2V_i^s - 1) \left[\sum_{j=1}^{n} V_j^{s'} (2V_j^s - 1) \right] \triangleq H_i^{s'} \quad . \tag{2.3}$$

Il valor medio della quantità tra parentesi quadre è zero a meno che non sia $s = s'$ nel qual caso tale valor medio è $\frac{n}{2}$.
A causa di questa pseuodoortogonalità si ha

$$\langle H_i^{s'} \rangle \approx (2V_i^{s'} - 1)\frac{n}{2} \tag{2.4}$$

e tale quantità è positiva se $V_i^{s'} = 1$ e negativa se $V_i^{s'} = 0$. Quindi, trascurando l'errore associato all'approssimazione introdotta per i termini relativi agli $s \neq s'$, lo stato s' può essere considerato stabile.
Definita la seguente funzione di Lyapunov

$$E = -\frac{1}{2}\mathbf{V}^\top \mathbf{T}\mathbf{V} = -\frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} T_{ij} V_i V_j \quad , \tag{2.5}$$

che chiameremo *energia*, si può dimostrare il seguente teorema:

Teorema 2.1. *La funzione E è decrescente.*

Dimostrazione. Essendo $\mathbf{T}$ simmetrica, la variazione ΔE dell'energia dovuta a una variazione ΔV_i dell'uscita del neurone i-esimo è data da

$$\Delta E = -\Delta V_i \sum_{j=1}^{n} T_{ij} V_j \quad . \tag{2.6}$$

Questo significa che se, per esempio, V_i cambia stato passando da 0 a 1, essendo $\Delta V_i = 1$ e $\sum_j T_{ij} V_j > 0$, la quantità ΔE è negativa; allo stesso risultato si arriva supponendo che V_i passi da 1 a 0 e si può concludere che E è una funzione decrescente. □

Come corollario, si verifica facilmente che quando il sistema raggiunge uno stato di equilibrio, cioè uno stato tale che $\Delta V_i = 0$ per ogni i, corrispondentemente si ha $\Delta E = 0$ e la funzione energia raggiunge un minimo.

Dato l'insieme S, dunque, è possibile costruire una rete per la quale gli elementi di S siano stati di equilibrio. Purtroppo non sono gli unici: simulazioni fatte dallo stesso Hopfield, hanno mostrato che l'algoritmo di memorizzazione (2.2) introduce stati stabili indesiderati (detti *spuri*) a causa dei quali la rete può fornire risposte sbagliate. Maggiore è il numero degli stati che si vogliono immagazzinare rispetto alle dimensioni della rete, cioè al numero n di neuroni, e maggiore è la probablità di errore; Hopfield trovò sperimentalmente che per valori di K superiori al valore limite di circa $0,15n$ la rete diventa *satura* e gli errori si verificano con una frequenza inaccettabile.

2.1.1 Capacità di una rete di Hopfield

La speranza matematica del valore di attivazione espresso dalla (2.4) è $\pm\frac{n}{2}$ a seconda del valore di $V_i^{s'}$ e i $(K-1)$ termini della (2.2) relativi agli $s \neq s'$ costituiscono un rumore di media nulla e varianza $\sigma^2 = \left[\frac{(K-1)n}{2}\right]$.
Considerando tale rumore gaussiano, approssimazione accettabile per $nK \gg 1$, si ricava la probabilità che si verifichi un errore in un singolo bit di una particolare memoria; essa è associata al verificarsi dell'evento $\langle H_i^{s'} \rangle = -\frac{n}{2}$ congiuntamente all'evento $H_j^{s'} = \sum_j T_{ij} V_j^{s'} > 0$ - o, analogamente, $\langle H_i^{s'} \rangle = \frac{n}{2}$ congiuntamente a $H_j^{s'} = \sum_j T_{ij} V_j^{s'} \leqslant 0$ - , ché in tal caso avviene una transizione indesiderata dovuta al rumore, e sarà (si veda la Figura 2.1):

$$P = \frac{1}{\sqrt{2\pi\sigma^2}} \int_{\frac{n}{2}}^{\infty} e^{-\frac{x^2}{2\sigma^2}} dx = \frac{1}{\sqrt{\pi}} \int_{\frac{1}{2}\sqrt{\frac{n}{K-1}}}^{\infty} e^{-t^2} dt \quad . \tag{2.7}$$

La (2.7) ci permette di calcolare la probabilità che non si verifichi alcun errore in alcuna delle n unità ovvero la probabilità di risposta corretta:

$$P_c = (1-P)^n \quad . \tag{2.8}$$

Tale quantità è funzione decrescente all'aumentare della cardinalità di S, e il massimo valore di K per cui essa rimane maggiore di $\frac{1}{2}$ potrebbe essere ragionevolmente interpretato come valore limite, cioè come misura della *capacità* della rete. Vediamo come calcolarla: vogliamo che sia

$$(1-P)^n \geqslant \frac{1}{2} \qquad \text{con} \quad P = \frac{1}{2}\left[1 - erf\left(\frac{1}{2}\sqrt{\frac{n}{K-1}}\right)\right] \tag{2.9}$$

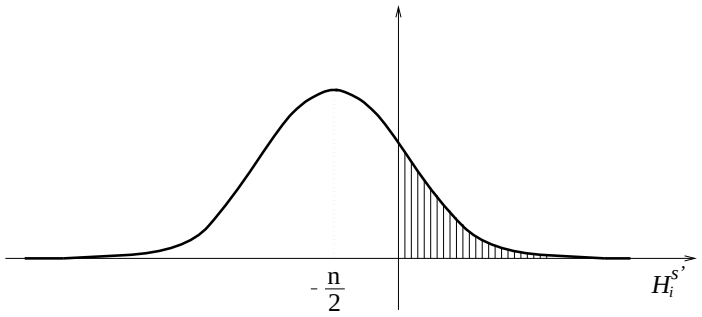


Figura 2.1: Distribuzione di probabilità del rumore.

ma siccome $P \ll 1$, $(1-P)^n \simeq e^{-nP}$ e dunque la disequazione che dobbiamo risolvere è $e^{-nP} \geqslant \frac{1}{2}$. Con semplici passaggi si ricava

$$K \leqslant \frac{n}{4\left[erf^{-1}\left(1-\frac{2\ln 2}{n}\right)\right]^2} + 1 \quad . \tag{2.10}$$

Dalla (2.10), con l'utilizzo del programma *Matlab*®, è stato ricavato per tentativi, per alcuni valori di K, il minimo valore di n che permette di avere una capacità, definita come sopra, pari a K. I risultati sono riportati nella seguente tabella.

K	5	6	7	9	10	15	20	30	40	50	100
n_{min}	34	48	63	96	113	206	307	525	758	1003	2336

Tabella 2.1: Numero minimo di neuroni necessario per avere una capacità pari a K.

Osservazione 2.1. Procedendo in modo analogo, si può trovare il valore massimo di K che limita l'errore a una quantità ϵ arbitrariamente piccola [14]; con l'approssimazione

$$efr(z) \simeq \frac{1}{2\pi}\frac{e^{-z^2}}{z} \quad , \tag{2.11}$$

si ricava

$$K_{max} \simeq \frac{1}{4 \ln I + 2 \ln \left(\frac{1}{\epsilon}\right)} \quad . \tag{2.12}$$

Osservazione 2.2. L'ipotesi che **T** sia simmetrica è fondamentale per ottenere una dimostrazione rigorosa del carattere non decrescente di E ma anche senza tale ipotesi il sistema si comporta come una memoria associativa; con simulazioni fatte utilizzando il metodo di Monte Carlo, infatti, è stato osservato che nei casi in cui non veniva raggiunto uno stato stabile, il piú delle volte la rete oscillava tra due stati o percorreva una traiettoria circolare tra stati vicini. Questo fatto può essere spiegato separando l'equazione (2.5) in due addendi dei quali uno è sempre negativo e l'altro è identico al primo se **T** è simmetrica ed ha media nulla se T_{ij} e T_{ji} sono scelti a caso.

Osservazione 2.3. Per due neuroni naturali p e q non sempre vi sono entrambe le sinapsi $p \to q$ e $q \to p$ e ci si può chiedere quale sia il comportamento di una RNA di Hopfield la cui matrice delle connessioni abbia qualche elemento nullo senza che lo sia il suo simmetrico (cioè $T_{ij} = 0, T_{ji} \neq 0$). Anche in questo caso il sistema continua a comportarsi come una memoria associativa ma la probabilità di errore aumenta.

2.2 Il modello continuo

In virtú del suo carattere binario, il neurone di McCulloch e Pitts è facile da analizzare ma non è un buon modello dei neuroni biologici. In questi ultimi, infatti, le informazioni sono codificate tramite una modulazione di posizione degli impulsi inviati lungo l'assone con la quale ogni cellula sopperisce all'incapacità di generare una risposta continua. In sostanza, con le dovute limitazioni dei sistemi fisici, ogni punto dell'intervallo $[0, 1]$ è messo in relazione con la frequenza media a breve termine alla quale il soma scatena il potenziale d'azione [3].
Nel primo modello di RN proposto da Hopfield non si tiene conto di questo fatto né, d'altro canto, del ritardo di propagazione dei potenziali d'azione lungo l'assone che è invece presente in tutti i sistemi fisici. Sulla base di

queste considerazioni, nel 1984 Hopfield propose un nuovo modello nel quale l'uscita di ciascuna unità di processo può assumere qualunque valore in $[0,1]$ e l'equazione che descrive la variazione nel tempo del valore di attivazione coinvolge operatori differenziali [15]. Inoltre dimostrò che le proprietà di memoria associativa, caratteristiche del modello discreto e asincrono, valgono anche per quello continuo e sincrono.
Sia, dunque, $V_i \in [0,1]$ l'uscita dell'i-esimo neurone, funzione monotona crescente del valore di attivazione u_i:

$$V_i = g_i(u_i) \qquad \text{o anche} \qquad u_i = g_i^{-1}(V_i) \tag{2.13}$$

dove g è la funzione sigmoide di Figura 1.3(b).
Per descrivere completamente il comportamento del neurone si deve aggiungere alla (2.13) l'equazione che definisce il valore di attivazione; in questo caso, esso non è piú la semplice somma algebrica degli ingressi ma è una soluzione dell'equazione differenziale

$$C_i \frac{du_i}{dt} = \sum_{j=1}^{n} T_{ij} V_j - \frac{u_i}{R_i} + I_i \quad , \tag{2.14}$$

dove C_i e R_i rappresentano la capacità e la resistenza di transmembrana e I_i è un valore costante che rappresenta una *polarizzazione* correlata agli stimoli esterni*. Un sistema fisico retto dalle stesse (2.13) e (2.14) è il circuito elettronico di Figura 2.2.

*La (2.14) riprende l'equazione, ben nota a chi si occupa di bioingegneria,

$$I_m = \frac{V'}{R_m} + C_m \frac{dV'}{dt}$$

che descrive la variazione nel tempo della differenza di potenziale V' tra l'interno e l'esterno della membrana cellulare, in risposta a uno stimolo di corrente I_m. Negli esperimenti di Hodgkin e Huxley, che per primi interpretarono la membrana cellulare come un circuito elettrico, I_m era iniettata dall'operatore, ma in realtà corrisponde alla somma dei segnali provenienti dall'esterno e dagli altri neuroni identificati nella (2.14) da I_i e da $\sum_{j=1}^{n} T_{ij} V_j$. R_m e C_m tengono conto, rispettivamente, della resistenza presentata dalla membrana cellulare al passaggio di ioni, cioè alla corrente, e dell'accumulo di cariche da parti opposte della membrana generato dalla pompa Na-Cl, elementi che permettono la trasmissione dell'informazione lungo l'assone ma che al tempo stesso ne costituiscono un fattore di ritardo.

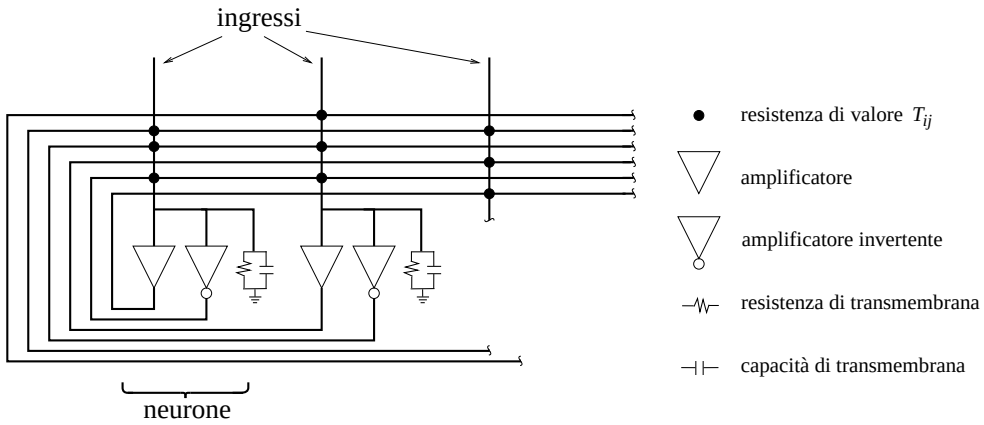


Figura 2.2: Il circuito elettrico di Hopfield.

Si consideri, ora, la funzione

$$E = -\frac{1}{2}\mathbf{V}^\top \mathbf{T}\mathbf{V} + \sum_{i=1}^{n} \frac{1}{R_i} \int_0^{V_i} g_i^{-1}(v)dv - \mathbf{V}^\top \mathbf{I} \tag{2.15}$$

dove $\mathbf{I} = (I_1, \ldots, I_n)^\top$ è il vettore delle polarizzazioni.

Teorema 2.2. *La funzione E decresce col tempo.*

Dimostrazione. Nell'ipotesi di simmetria della $\mathbf{T}$, si ha

$$\frac{dE}{dt} = \sum_{i=1}^{n} \frac{dE}{dV_i}\frac{dV_i}{dt} = \sum_{i=1}^{n} \frac{dV_i}{dt}\Big(-\mathbf{T}\mathbf{V} + \frac{u_i}{R_i} - I_i\Big) \quad . \tag{2.16}$$

Sostituendo alla quantità tra parentesi il primo membro della (2.14) si ricava

$$\frac{dE}{dt} = -\sum_{i=1}^{n} \frac{dV_i}{dt}\frac{du_i}{dt}C_i = -\sum_{i=1}^{n} C_i\left(\frac{dV_i}{dt}\right)^2 \frac{du_i}{dV_i} \tag{2.17}$$

dove l'ultima derivata può essere esplicitata in base alla (2.13) ottenendo

$$\frac{dE}{dt} = -\sum_{i=1}^{n} C_i\left(\frac{dV_i}{dt}\right)^2 \left[\frac{d}{dV_i} g_i^{-1}(V_i)\right] \quad . \tag{2.18}$$

Siccome g_i^{-1} è monotona crescente, il termine tra parentesi quadre è sempre positivo, come positiva è anche C_i; a questo punto è facile concludere che $\frac{dE}{dt} \leqslant 0$. □

In particolare vale la seguente doppia implicazione:

$$\frac{dE}{dt} = 0 \qquad \Longleftrightarrow \qquad \frac{dV_i}{dt} = 0 \quad \forall\, i \tag{2.19}$$

cioè il valore dell'energia non cambia nel tempo se e solo se la rete ha raggiunto uno stato di equilibrio.
A volte si utilizza come funzione non lineare una tangente iperbolica anziché una sigmoide e allora V_i assume valori nell'intervallo $(-1, 1)$ anziché nell'intervallo $(0, 1)$; in questo caso il teorema appena dimostrato continua a valere, poggiando la dimostrazione della decrescenza di E sulla monotonia e crescenza di g_i^{-1}, condizione verificata anche dalla tangente iperbolica.

2.2.1 Corrispondenza tra i due modelli

Si metterà ora in luce la relazione che c'è tra i punti di equilibrio del sistema descritto dalle (2.13) e (2.14) e quelli del modello discreto (descritto dalla (2.1)). Se nella (2.15) si pone $I_i = 0$ e $T_{ii} = 0 \;\; \forall\, i$, essa diventa uguale alla somma di due termini, il primo dei quali coincide con la(2.5) mentre il secondo è:

$$\sum_{i=1}^{n} \frac{1}{R_i} \int_0^{V_i} g_i^{-1}(v)dv \quad . \tag{2.20}$$

Se adesso si modifica la (2.13) introducendo un guadagno,

$$V_i = g_i(\lambda u_i) \qquad \text{o anche} \qquad u_i = \frac{1}{\lambda} g_i^{-1}(V_i) \quad , \tag{2.21}$$

la (2.20) diventa

$$\frac{1}{\lambda} \sum_{i=1}^{n} \frac{1}{R_i} \int_0^{V_i} g_i^{-1}(v)dv \tag{2.22}$$

e per λ molto grande diventa trascurabile. Allora, se come funzioni g_i si scelgono sigmoidi con guadagni elevati, cioè con un andamento simile al gradino (vedi Figura 2.3), le funzioni energetiche dei due modelli differiscono per un termine trascurabile. Inoltre, poiché la restrizione di E a uno spigolo dell'ipercubo unitario definito da $\left\{\mathbf{V} = (V_1, \ldots, V_n) \;:\; 0 \leqslant V_i \leqslant 1 \;\forall\, i = 1, \ldots, n\right\}$ è una funzione lineare di un singolo V_i, i minimi energetici del modello continuo coincidono con quelli del modello discreto.

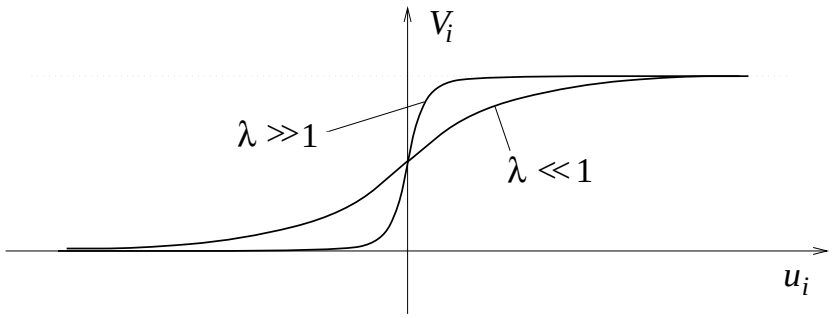


Figura 2.3: Variazione della sigmoide in funzione di λ.

2.3 Il sistema di Li, Michel e Porod

In un articolo del 1988 [16] Jian Hua Li, Anthony Michel e Wolfgang Porod presentarono lo studio di una nuova classe di RN il cui stato $\mathbf{x} \in \Omega$ soddisfa l'equazione differenziale non lineare

$$\frac{d\mathbf{x}}{dt} = -\mathbf{H}(\mathbf{x})(-\mathbf{T}\mathbf{x} + \mathbf{S}(\mathbf{x}) - \mathbf{I}) \tag{2.23}$$

dove $\mathbf{x} \in (-1,1)^n$,
$\mathbf{H}$ è una funzione $(-1,1)^n \to \mathbb{R}^{n\times n}$ cioè $\forall\, \mathbf{x}$ $\mathbf{H}(\mathbf{x})$ è una matrice $n \times n$,
$\mathbf{T}$ è una matrice $n \times n$ costante,
$\mathbf{S}(\mathbf{x}) = (S_1(x_1), \ldots, S_n(x_n))^\top$ con $S_i : (-1,1) \to \mathbb{R}$
e $\mathbf{I} = (I_1, \ldots, I_n)^\top$ è un vettore costante.
Essi associarono al sistema (2.23) la funzione energia $E : (-1,1)^n \to \mathbb{R}$ definita da

$$E(\mathbf{x}) = -\frac{1}{2}\mathbf{x}^\top\mathbf{T}\mathbf{x} + \sum_{i=1}^{n}\int_0^{x_i} S_i(\rho)d\rho - \mathbf{x}^\top\mathbf{I} \tag{2.24}$$

e dimostrarono il seguente teorema:

Teorema 2.3. *Se* $\forall\, \mathbf{x} \in (-1,1)^n$ $\mathbf{H}(\mathbf{x})$ *è simmetrica e definita positiva,* $\mathbf{T}$ *è simmetrica e* $\forall\, i : 1 \leqslant i \leqslant n$ $S_i(0) = 0$, S_i *è monotona crescente, dispari, di classe* $\mathcal{C}^1$, *invertibile, se l'inversa è di classe* $\mathcal{C}^1$ *e se esiste* $\dfrac{d^2S_i(x_i)}{d^2x_i}$, *allora* $\mathbf{x} \in (-1,1)^n$ *è un punto di equilibrio per* (2.23) *se e solo se* $\nabla E(\mathbf{x}) = 0$, *dove* ∇ *rappresenta l'operatore gradiente.*

Inoltre dimostrarono che sotto certe ipotesi, quasi sempre verificate nei problemi reali, $\mathbf{x}$ è un punto di equilibrio stabile per (2.23) se e solo se è anche un punto di minimo per E.
Come fecero notare gli stessi autori se nella (2.23) si operano le sostituzioni

$$\mathbf{x} = \mathbf{V} \quad , \qquad S_i(\mathbf{x}) = \frac{g_i^{-1}(V_i)}{\lambda R_i} \quad , \tag{2.25}$$

$$H_{ij}(\mathbf{x}) = 0 \quad \text{per } i \neq j \qquad \text{e} \qquad H_{ii}(\mathbf{x}) = \frac{\lambda}{C_i \dfrac{d}{dV_i} g_i^{-1}(V_i)} \quad , \tag{2.26}$$

si ottiene un sistema di equazioni differenziali la cui generica equazione ha la forma

$$\frac{dV_i}{dt} = \frac{\lambda}{C_i \dfrac{d}{dV_i} g_i^{-1}(V_i)} \left(\sum_{j=1}^{n} T_{ij} V_j - \frac{g_i^{-1}(V_i)}{\lambda R_i} + I_i \right) \quad . \tag{2.27}$$

Essendo $\frac{1}{\lambda} g_i^{-1}(V_i) = u_i$, dalla (2.27) si ricava

$$C_i \frac{du_i}{dV_i} \frac{dV_i}{dt} = \sum_{j=1}^{n} T_{ij} V_j - \frac{u_i}{R_i} + I_i \tag{2.28}$$

che è uguale alla (2.14) di Hopfield. Con le sostituzioni (2.25) e (2.26) la (2.24) diventa

$$E(\mathbf{V}) = -\frac{1}{2} \mathbf{V}^\top \mathbf{T} \mathbf{V} + \sum_{i=1}^{n} \frac{1}{\lambda R_i} \int_0^{V_i} g_i^{-1}(v) dv - \mathbf{V}^\top \mathbf{I} \tag{2.29}$$

che altro non è che la (2.15) modificata tenendo conto del guadagno λ.
In un loro successivo lavoro [17], Li, Michel e Porod studiarono un altro sistema lineare molto semplice

$$\frac{d\mathbf{x}}{dt} = \mathbf{T}\mathbf{x} + \mathbf{I} \qquad \text{con la restrizione} \quad -1 \leqslant x_i \leqslant 1 \quad , \tag{2.30}$$

e dimostrarono che, se $\mathbf{T}$ è simmetrica e non singolare, allora c'è una corrispondenza biunivoca tra i punti asintoticamente stabili di (2.30) e i punti di minimo della funzione

$$E = -\frac{1}{2} \mathbf{x}^\top \mathbf{T} \mathbf{x} - \mathbf{x}^\top \mathbf{I} \quad . \tag{2.31}$$

Ora, siccome per $\lambda \to \infty$ la (2.29) tende alla (2.31), si può concludere che, per i sistemi con quadagno elevato, nell'intorno di un punto asintoticamente stabile di (2.30) c'è un punto asintoticamente stabile di (2.28). Nei casi pratici, è molto probabile che la corrispondenza tra questi due insiemi di punti sia un'identità e pertanto tutti i risultati ottenuti per il sistema (2.30) potranno essere ritenuti validi anche per le reti di Hopfield.

2.4 Reti di Hopfield per problemi di ottimizzazione combinatoria

Il risultato piú interessante che emerge dai paragrafi precedenti è la corrispondenza tra i punti di minimo della funzione energia e gli stati di equilibrio della rete; questa corrispondenza può essere sfruttata in un campo diverso da quello della memorizzazione associativa ovvero per la risoluzione di problemi di ottimizzazione combinatoria (OC).

2.4.1 Cenni di teoria della complessità

Prima di vedere nel dettaglio come si possano risolvere i problemi di OC, si ritiene opportuno definire alcuni concetti che spesso verranno richiamati nel seguito; per una trattazione piú ampia della teoria che si occupa della complessità dei problemi e delle caratteristiche degli algoritmi che li risolvono, si rimanda a [18].

Definizione 2.1. *Si definisce* problema*, e si indica con* Π*, una generica domanda che solitamente possiede diversi* parametri*, o variabili libere, i cui valori non sono specificati. Un problema consiste in:*

- *una descrizione generale di tutti i parametri*
- *una richiesta delle proprietà che la risposta, o* soluzione*, deve soddisfare.*

Definizione 2.2. *Un caso particolare, o* istanza*,* $\mathcal{I}$ *del problema è una specificazione dei valori assunti dai parametri del problema.*

Definizione 2.3. *Se esiste un algoritmo che risolve un problema* Π*, la* lunghezza di ingresso *per un'istanza* $\mathcal{I}$ *di* Π *è il numero di simboli che servono per descrivere* $\mathcal{I}$ *nel linguaggio di ingresso dell'algoritmo.*

Definizione 2.4. *La* funzione di complessità temporale *per un algoritmo, esprime il tempo di computazione fornendo, per ogni possibile lunghezza* n *di ingresso, la massima quantità di tempo richiesta dall'algoritmo per risolvere un'istanza del problema di lunghezza* n*.*

Definizione 2.5. *Una funzione* $f(n)$ *è* $O(g(n))$ *se esiste una costante* c *tale che* $|f(n)| \leqslant c\,|g(n)| \quad \forall\, n \geqslant 0$*.*

Definizione 2.6. *Un algoritmo* polinomiale nel tempo *è un algoritmo la cui funzione di complessità temporale è* $O(p(n))$ *per qualche funzione polinomiale* p*. Ogni algoritmo non temporalmente polinomiale è detto* esponenziale *(nel tempo).*

Definizione 2.7. *Un problema* Π *è detto* intrattabile *se non si conosce nessun algoritmo temporalmente polinomiale che lo risolve.*

Definizione 2.8. *Si definisce P (polinomiale) la classe dei problemi* Π *per i quali esiste una macchina di Turing deterministica (MTD) che esegue un algoritmo temporalmente polinomiale che risolve* Π*.*

Definizione 2.9. *Si definisce NP (polinomiale nondeterministica) la classe dei problemi* Π *per i quali esiste una MTD che, data un'istanza del problema e fornita una soluzione, esegue un algoritmo temporalmente polinomiale che verifica l'esattezza della soluzione; in pratica, si suppone di avere già una soluzione, che si congettura essere esatta, e di doverla semplicemente verificare (una tale MT si dice nondeterministica (MTND)). Si può dimostrare che se* $\Pi \in$ *NP allora esiste un polinomio* p *tale che* Π *può essere risolto da una MT in un numero di passi* $O(2^{p(n)})$*.*

Definizione 2.10. *Un problema* $\Pi \in$ *NP è detto NP-C (NP completo) se* $\forall\, \Pi' \in$ *NP esiste una MTD che traduce in tempo polinomiale ogni istanza di* Π' *in un'istanza di* Π*.*

Alla luce di queste precisazioni, diventa piú chiaro il significato che gli aggettivi 'facile' e 'difficile' assumono nell'ambito della teoria della complessità e cosa si intenda per 'poco' o 'tanto' tempo, concetti che avevamo anticipato nell'introduzione.

Nota 2.1. La storia di questa nuova area dell'indagine matematica è stata accompagnata, fin dall'inizio, dal sospetto che le classi P e NP siano diverse; tale ipotesi, non ancora dimostrata né smentita, va sotto il nome di '*congettura* $P \neq NP$' e costituisce uno dei tanti quesiti che attendono risposta.

2.4.2 Considerazioni generali

L'idea di fondo che consente l'utilizzo delle RN per la risoluzione di problemi di OC è la seguente: se si riesce a codificare con uno stato $\mathbf{x}$ della rete ogni soluzione del problema, basterà costruire in maniera opportuna dall'istanza del problema gli elementi delle matrici $\mathbf{T}$ e $\mathbf{I}$ in modo che il processo di minimizzazione dell'energia coincida con la deriva di $\mathbf{x}$ verso una soluzione; questo si può fare associando l'energia della rete alla distanza tra la soluzione ottima, definita implicitamente dall'istanza del problema, e la generica soluzione fornita dalla decodifica dello stato istantaneo della rete.
Il motivo per cui questo metodo euristico, diverso dai tradizionali algoritmi, riscuote notevole successo, pur non fornendo sempre la soluzione ottima, è da ricercarsi nell'elevata velocità di risposta.
Nei problemi pratici, molto spesso è preferibile avere una risposta buona in poco tempo che la risposta ottima in un tempo troppo lungo; le RN, data la quantità di connessioni e di celle elementari che contengono, riducono i tempi di calcolo. Detto altrimenti, la complessità temporale di un problema è in parte trasformata nella complessità strutturale di un sistema che compie molte operazioni contemporaneamente.
L'inconveniente dell'approccio neurale ai problemi di natura combinatoria è che non si ha nessuna garanzia né sulla qualità della soluzione trovata né sul tempo necessario per ottenerla. Inoltre si può dimostrare [19] che la potenza di calcolo di una RN non è superiore a quella di una normale MTD; non è possibile, cioè risolvere un problema NP in un numero di passi polinomiale. Tuttavia le RN, grazie alla loro complessità strutturale, ad ogni passo com-

piono un numero di operazioni elementari dell'ordine di n^2 cioè del numero di connessioni e anche se in teoria questo non ci permette di considerare semplice un problema intrattabile, agli effetti pratici l'utilizzo dei sistemi neuromorfici comporta spesso notevoli vantaggi temporali.

2.4.3 Il problema del commesso viaggiatore

La prima rete per OC risolveva il problema del commesso viaggiatore (noto anche nella forma acrostica *TSP* dell'inglese *travelling salesman problem*) analizzato nel 1985 dallo stesso Hopfield e da David Tank [20]. Esso è di facile descrizione ed è un buon esempio di un problema NP-C. Per questi motivi è diventato il problema esemplare per chi studia le possibilità di migliorare le prestazioni di una RN di Hopfield e ad esso sarà dedicato questo paragrafo rimandando al successivo una trattazione sommaria delle altre applicazioni. Sia dato un insieme di città $\{A, B, C, \dots\}$ e si associ a ciascuna coppia di città una distanza: $d_{AB}, d_{AC}, \dots, d_{BC}, \dots$. Il problema è quello di trovare un percorso chiuso che tocchi una e una sola volta ciascuna città e che abbia una lunghezza totale minima (o piú piccola di una lunghezza massima). Ogni percorso è definito da una successione delle città e la sua lunghezza totale è la somma delle distanze associate a ciascuna coppia di città adiacenti:

$$d_{(B,F,E,G,\dots,W)} = d_{BF} + d_{FE} + d_{EG} + \dots + d_{WB} \tag{2.32}$$

La soluzione di un TSP di dimensione m consiste in una lista ordinata delle m città e per mappare ogni soluzione in uno stato della rete si deve costruire uno schema rappresentativo che consenta di decodificare la lista dalle uscite digitali dei neuroni.
Hopfield scelse una rappresentazione in cui la posizione finale di ciascuna città nella lista è specificata da un insieme di m neuroni tutti con uscita nulla tranne quello che corrisponde alla posizione della città nella lista. Per esempio, in un problema di 10 città, quella che nella lista occupa la 6^a posizione sarà codificata dalla 10-upla $(0,0,0,0,0,1,0,0,0,0)$. Per ciascuna delle m città ci sarà bisogno di m neuroni e cioè di un totale di $n = m^2$ neuroni.
Per rappresentare la generica soluzione, è conveniente utilizzare una scrittura matriciale in cui ogni riga corrisponde a una città e ogni colonna alla

posizione nella lista. In un TSP di dimensione 5, per esempio, lo stato

	1	2	3	4	5
A	0	1	0	0	0
B	0	0	0	1	0
C	1	0	0	0	0
D	0	0	0	0	1
E	0	0	1	0	0

(2.33)

corrisponde al ciclo $C \to A \to E \to B \to D \to C$ e la sua lunghezza totale è $d_{CA}+d_{AE}+d_{EB}+d_{BD}+d_{DC}$. Per rispettare questa rappresentazione, i simboli che indicano le uscite dei neuroni avranno due indici (V_{Xj}) il primo dei quali corrisponde al nome della città ($A, B, \dots$) e il secondo alla posizione nella lista ($1, 2, \dots$); nel caso dell'esempio precedente $V_{A1} = 0$, $V_{A2} = 1$, $V_{B1} = 0$ e cosí via.

Si devono, ora, definire le matrici $\mathbf{T}$ e $\mathbf{I}$ in modo che l'energia della rete sia una funzione *costo* che assume valori piccoli in corrispondenza di buone soluzioni al TSP. A tal proposito, si deve evitare che la rete finisca in stati che non rappresentano soluzioni ammissibili del problema; essi sono quegli stati la cui rappresentazione matriciale ha piú di un 1 sulla stessa riga - in tal caso la città verrebbe visitata piú di una volta - o sulla stessa colonna - in tal caso ci sarebbero due città visitate contemporaneamente - e quelli che hanno una riga o una colonna di soli zeri. Hopfield propose la seguente funzione:

$$E = \frac{A}{2}\sum_X\sum_i\sum_{j\neq i} V_{Xi}V_{Xj} + \frac{B}{2}\sum_i\sum_X\sum_{X\neq Y} V_{Xi}V_{Yi} + \\ + \frac{C}{2}\left(\sum_X\sum_i V_{Xi} - n\right)^2 + \frac{D}{2}\sum_X\sum_{X\neq Y}\sum_i d_{XY}V_{Xi}\big(V_{Y,i+1} + V_{Y,i-1}\big). \quad (2.34)$$

dove $A, B, C, D > 0$. La prima sommatoria tripla è nulla se e solo se ogni riga contiene al massimo un 1. La seconda è nulla se e solo se ogni colonna contiene al massimo un 1. Il terzo termine è zero se e solo se ci sono n elementi della rappresentazione matriciale uguali a 1. Con questi tre accorgimenti, se A, B e C hanno valori opportuni, si privilegiano gli stati che corrispondono a soluzioni ammissibili associando loro un'energia minore. Il quarto termine,

infine, è relativo alla lunghezza del percorso.
A questo punto è facile ricavare **T** e **I** dalla (2.34):

$$\begin{aligned} T_{XiYj} &= -A\delta_{XY}(1-\delta_{ij}) - B\delta_{ij}(1-\delta_{XY}) - C - Dd_{XY}(\delta_{j,i+1} + \delta_{j,i-1}) \\ I_{Xi} &= Cn \end{aligned} \tag{2.35}$$

dove, $\delta_{ij} = 1$ se $i = j$, 0 altrimenti. Per scegliere i parametri A, B, C e D bisogna cercare un compromesso tra il criterio delle soluzioni ammissibili e quello della minima lunghezza; il primo, infatti, richiede un valore elevato delle costanti A, B e C, ma queste non possono essere troppo grandi rispetto a D perché in tal caso la qualità delle soluzioni rischia di essere pessima.
Si noti, inoltre, che non è necessario che ogni punto della traiettoria percorsa da **V** in Ω sia una matrice di permutazione avente la forma espressa dalla (2.33), basta che lo sia lo stato finale; anzi, il tentativo, fatto dallo stesso Hopfield, di utilizzare reti il cui spazio delle decisioni consiste dei soli vertici dell'ipercubo ha portato a soluzioni di poco migliori di percorsi casuali ed egli ne concluse che la possibilità di attraversare tutto l'ipercubo unitario $[0,1]^n$ aumenta le capacità computazionali della rete.

2.4.4 Altre applicazioni

Alla prima idea di Hopfield, ne seguirono ben presto molte altre che hanno provato la versatilità delle RN e la vastità del loro campo di applicazione. Ne descriveremo brevemente le piú note.

Il convertitore Analogico/Digitale

Un convertitore analogico/digitale a 4 bit (vedi Figura 2.4) è costituito da quattro amplificatori le cui tensioni di uscita sono decodificate per ottenere la parola binaria richiesta, una rete di resistenze di reazione, un insieme di resistori che alimentano gli ingressi degli amplificatori con correnti costanti di valori diversi e un altro insieme di resistori che iniettano, negli stessi ingressi degli amplificatori, correnti proporzionali al valore analogico x che si vuole convertire. Il circuito è costruito in modo che valga l'approssimazione

$$\sum_{i=0}^{3} V_i 2^i \approx x \quad . \tag{2.36}$$

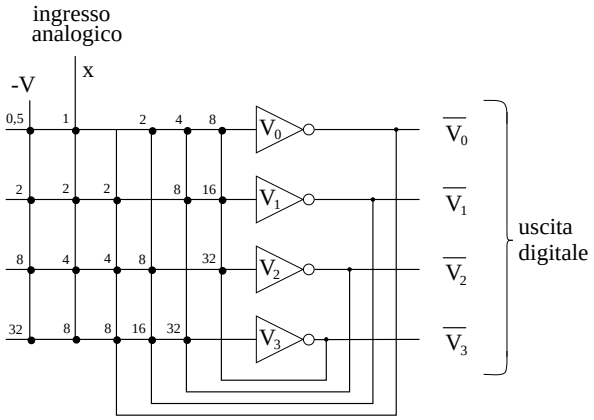


Figura 2.4: Struttura di un convertitore analogico/digitale.

Se la parola $V_3V_2V_1V_0$ è la rappresentazione digitale ottima di x, allora devono essere verificate due condizioni; la prima è che ogni V_i abbia valore 0 o 1 (o almeno sia abbastanza vicino a questi due valori da permettere al circuito comparatore di stabilire livelli digitali), la seconda è che in corrispondenza a $V_3V_2V_1V_0$ la funzione

$$E = \frac{1}{2}\big(x - \sum_{i=0}^{3} V_i 2^i\big)^2 \tag{2.37}$$

abbia un minimo [21]. La (2.37), comunque, non assicura che i punti di minimo di E stiano sui vertici dell'ipercubo perché la diagonale della matrice $\mathbf{T}$ che se ne ricava non è identicamente nulla; per ovviare a questo problema basta aggiungere un termine addizionale della forma

$$-\frac{1}{2}\sum_{i=0}^{3}(2^i)^2\big[V_i(V_i-1)\big] \tag{2.38}$$

ricavando

$$E = -\frac{1}{2}\sum_{i=0}^{3}\sum_{\substack{j=0\\ j\neq i}}^{3}(-2^{i+j})V_iV_j - \sum_{i=0}^{3}(-2^{2i-1}+2^i x)V_i \tag{2.39}$$

e, automaticamente,

$$T_{ij} = -2^{i+j} \qquad\qquad I_i = (-2^{2i-1}+2^i x) \quad . \tag{2.40}$$

Il problema della decomposizione

Si tratta di rilevare la presenza di una forma d'onda nota in un segnale composto anche da altre forme d'onda e corrotto dal rumore.
Si immagini, per esempio, di avere un segnale risultante dalla somma di impulsi gaussiani stereotipati, di ampiezza e istante centrale incogniti, e di volerlo scomporre nelle sue componenti; per essere piú precisi, si supponga di disporre di N campioni analogici del segnale, registrati negli istanti $t = 1, 2, \ldots, N$ e che le funzioni gaussiane siano della forma

$$\epsilon_{\sigma t}(i) = e^{-(\frac{i-t}{\sigma})^2} \tag{2.41}$$

dove σ può assumere valori in un insieme finito e t può essere un qualunque istante di campionamento. A ciascuna coppia (σ, t) si associa un neurone $V_{\sigma t}$ la cui uscita è 1 se si rileva la presenza del segnale cercato, 0 altrimenti.
Una funzione energia che raggiunge il suo minimo quando è risolto il problema decisionale è [21]

$$E = -\frac{1}{2}\sum_{i=1}^{N}\Big(x_i - \sum_{\sigma=\sigma_1}^{\sigma_{max}}\sum_{t=1}^{N} V_{\sigma t}\epsilon_{\sigma t}(i)\Big)^2 - \frac{1}{2}\sum_{i=1}^{N}\sum_{\sigma=\sigma 1}^{\sigma_{max}}\sum_{t=1}^{N}(\epsilon_{\sigma t}(i))^2\left[V_{\sigma t}(V_{\sigma t}-1)\right] \tag{2.42}$$

da cui si ricava

$$T_{\sigma t,\sigma' t'} = \sum_{i=1}^{N} e^{-[(\frac{i-t}{\sigma})^2+(\frac{i-t'}{\sigma'})^2]} \tag{2.43}$$

$$I_{\sigma t} = \sum_{i=1}^{N} x_i e^{-(\frac{i-t}{\sigma})^2} + \sum_{i=1}^{N} e^{-2(\frac{i-t}{\sigma})^2} \quad . \tag{2.44}$$

La programmazione lineare

Consiste nel cercare di render minima una funzione costo

$$\pi = \vec{A} \bullet \vec{V} \tag{2.45}$$

dove $\vec{A}$ è un vettore N-dimensionale di coefficienti per le N variabili che sono le componenti di $\vec{V}$; queste ultime sono soggette a M vincoli lineari:

$$\vec{D_j} \bullet \vec{V} \geqslant B_j \qquad \text{con} \quad \vec{D_j} = (D_{j1}, \ldots, D_{jN})^\top \quad j = 1, \ldots, M \quad . \tag{2.46}$$

Sebbene non si conosca un metodo diretto per costruire una funzione come la (2.31), si può dimostrare che il circuito di Figura 2.5 fornisce la soluzione a questo problema di OC. Le N uscite (V_i) degli amplificatori di sinistra e

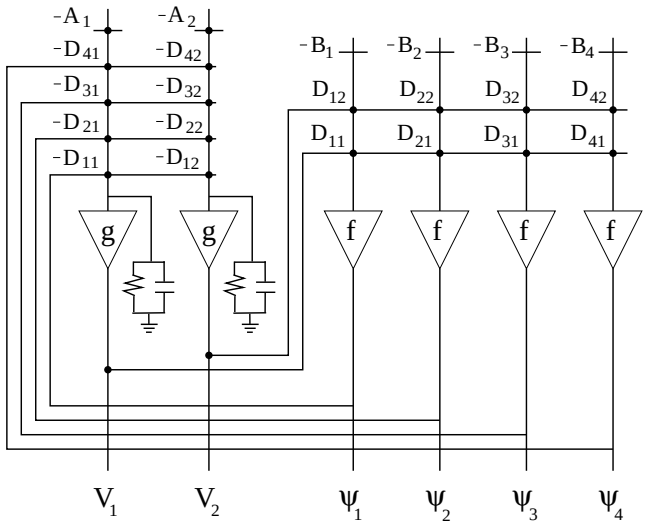


Figura 2.5: Una rete per la risoluzione di un problema di programmazione lineare con 2 variabili e 4 costanti.

le M uscite (ψ_i) di quelli di destra rappresentano, rispettivamente, i valori delle variabili e i vincoli [21].
Le relazioni ingresso-uscita degli amplificatori di sinistra (contrassegnati dalla lettera g) sono delle sigmoidi mentre per gli amplificatori di destra (contrassegnati dalla lettera f) valgono

$$u_j = \vec{D_j} \bullet \vec{V} - B_j \quad , \tag{2.47}$$

$$\psi_j = f(u_j) \qquad \text{dove} \quad f(z) = \begin{cases} 0 & \text{se } z \geqslant 0 \\ -z & \text{se } z < 0 \end{cases} . \tag{2.48}$$

La (2.48) fa in modo che l'uscita degli amplificatori ψ sia un valore positivo elevato quando non è rispettato il vincolo corrispondente. Se si fa l'ipotesi che il tempo di risposta di questi amplificatori sia trascurabile in confronto a

quello degli amplificatori di sinistra, allora si può ritenere valida le seguente equazione:

$$C_i \frac{du_i}{dt} = -A_i - \frac{u_i}{R} - \sum_j D_{ij} f(u_i) \quad . \tag{2.49}$$

Si consideri, ora, la funzione energia

$$E = (\vec{a} \bullet \vec{V}) + \sum_j F(\vec{D_j} \bullet \vec{V} - B_j) + \sum_i \frac{1}{R} \int_0^{V_i} g^{-1}(V) dV \tag{2.50}$$

dove F è definito implicitamente dalla

$$f(z) = \frac{dF(z)}{dz} \quad ; \tag{2.51}$$

la variazione di E rispetto al tempo è

$$\frac{dE}{dt} = \sum_i \frac{dV_i}{dt} \Big[\frac{u_i}{R} + A_i + \sum_j D_{ij} f(\vec{D_j} \bullet \vec{V} - B_j) \Big] \tag{2.52}$$

che facendo uso della (2.49) si può scrivere come

$$\frac{dE}{dt} = -\sum_i C_i \frac{dV_I}{dt} \frac{du_i}{dt} = -\sum_i C_i g^{-1}(V_i) \Big(\frac{dV_i}{dt} \Big)^2 \quad . \tag{2.53}$$

Siccome C_i è positivo e $g^{-1}(V_i)$ è monotona crescente, si ricava

$$\frac{dE}{dt} \leqslant 0 \qquad \text{e anche} \qquad \frac{dE}{dt} = 0 \iff \frac{dV_i}{dt} = 0 \quad \forall\, i \quad . \tag{2.54}$$

Il taglio di un grafo

Sia $\hat{G} = (\hat{V}, \hat{E})$ un grafo pesato unidirezionale, dove $\hat{V}$ denota l'insieme dei nodi di $\hat{G}$ e $\hat{E}$ l'insieme degli archi di $\hat{G}$. Si indichi con $\mathbf{T}$ la matrice simmetrica corrispondente ai pesi degli archi di $\hat{G}$. Sia $\hat{V}_1$ un sottoinsieme di $\hat{V}$ e sia $\hat{V}_{-1} = \hat{V} - \hat{V}_1$. L'insieme degli archi che incidono su un nodo in $\hat{V}_1$ e su un nodo in $\hat{V}_{-1}$ è detto *taglio* di $\hat{G}$. Il *peso* di un taglio è la somma dei pesi degli archi che lo compongono. Un *taglio minimo* di un grafo è un taglio con peso minimo.
Sia, ora, $N = (\mathbf{T}, \mathbf{I})$ una RN costituita da neuroni binari; per essa è possibile definire le seguenti tre quantità: W^{++}, ovvero la somma dei pesi degli archi

che collegano due nodi con uscita 1, W^{--} e W^{+-} definite in modo analogo. Sfruttando queste definizioni, è possibile riscrivere la (2.5) come

$$E = 2(W^{++} + W^{--} - W^{+-}) \tag{2.55}$$

$$E = 2(W^{++} + W^{--} + W^{+-}) - 4W^{+-} \quad . \tag{2.56}$$

Il primo termine della (2.56), è la somma dei pesi degli archi, dunque è costante; ne segue che E è massima quando W^{+-} è minima. Ma W^{+-} è il peso del taglio di N che separa l'insieme $\hat{V}_1$ dei nodi che hanno uscita 1 dall'insieme $\hat{V}_{-1}$ dei nodi che hanno uscita 0; dunque anche il problema del taglio minimo può essere messo in relazione con l'evoluzione di una RN [22].

I codici di Huffman

Si consideri una sorgente d'informazione S, stazionaria e senza memoria, con alfabeto $A = \{a_1, \ldots, a_K\}$. Ogni lettera a_i ha una probabilità a priori p_i di essere emessa dalla sorgente; si suppone, ovviamente, $p_i > 0$ e $\sum_i p_i = 1$ per la distribuzione di probabilità (DP) $P = \{p_1, \ldots, p_K\}$.
Il problema della codifica blocco-lunghezza variabile (B-LV) consiste nel realizzare una traduzione dalle lettere dell'alfabeto primario A alle parole di codice di un alfabeto secondario B (per esempio binario) tale che:

- il codice sia *univocamente decodificabile* (UD) cioè esista una sola sequenza di lettere di A per ogni sequenza di parole di codice
- le parole di codice $w_1, \ldots, w_K$ abbiano una lunghezza $l_1, \ldots, l_K$ tale che sia minima la lunghezza media

$$\bar{l} = \sum_{i=1}^{K} p_i l_i \quad . \tag{2.57}$$

Questo problema ha una soluzione ottima $l_1^*, \ldots, l_K^*$ che può essere trovata utilizzando l'algoritmo di Huffman [23].
Nel caso di un codice di Huffman binario, la disuguaglianza di Kraft

$$\sum_{i=1}^{K} 2^{-l_i^*} \leqslant 1 \quad , \tag{2.58}$$

valida per qualunque codice UD, diventa un'uguaglianza e ciò significa che le lunghezze ottime $l_1^*, \ldots, l_K^*$ sono tali che $2^{-l_1^*}, \ldots, 2^{-l_K^*}$ è una DP.
Il teorema fondamentale di Shannon sulla codifica di sorgente fornisce le seguenti limitazioni per $\bar{l}^* = \sum_{1=1}^{K} p_i l_i^*$:

$$H(S) \leqslant \bar{l}^* < H(S) + 1 \quad , \tag{2.59}$$

dove $H(S) = \sum_{i=1}^{K} p_i(-\log p_i)$ è l'*entropia* della sorgente. Il problema si riduce, allora, alla minimizzazione della quantità

$$\left[\bar{l} - H(S)\right]^2 \quad . \tag{2.60}$$

Per poter utilizzare le reti di Hopfield, i cui neuroni sono binari, si devono codificare la lunghezze $l_1, \ldots, l_K$ delle parole di codice nel modo suggerito dallo stesso Hopfield in [20]: sia $L = \max_i l_i$ e si associ ad ogni lettera a_i la L-upla $V_{i1}, \ldots, V_{iL}$ tale che, se la lunghezza della parola di codice w_i corrispondente ad a_i è l_i, si ha

$$\begin{cases} V_{ij} = 1 & \text{per } j = l_i \\ V_{ij} = 0 & \text{per } i \leqslant j \leqslant L \ , \ j \neq l_i \quad . \end{cases} \tag{2.61}$$

Come si può notare, ogni neurone ha due pedici, il primo ($X \in \{1, \ldots, K\}$) indica la parola e il secondo ($i \in \{1, \ldots, L\}$) indica la lunghezza; ad esempio la matrice

	1	2	3	$\cdots$	r	$\cdots$	L
a_1	0	1	0	$\cdots$	0	$\cdots$	0
a_2	0	0	1	$\cdots$	0	$\cdots$	0
$\vdots$							
a_K	0	0	0	$\cdots$	1	$\cdots$	0

(2.62)

identifica un codice che associa a a_1 una parola di lunghezza 2, ad a_2 una parola di lunghezza 3, ... , ad a_K una parola di lunghezza r. Con questa codifica, però, si deve assicurare che la rete fornisca soluzioni non ambigue e pertanto sarà necessario introdurre, nell'espressione dell'energia, i termini

- $\dfrac{C}{2} \sum_X \sum_i \sum_{j \neq i} V_{Xi} V_{Xj}$ $\quad (= 0 \iff$ c'è al massimo un 1 su ogni colonna)

- $\frac{D}{2}\Big[\sum_X \sum_i V_{Xi} - K\Big]^2 \quad (= 0 \iff$ ci sono K uscite pari a 1)
- $\frac{F}{2}\sum_X \sum_i V_{Xi}(V_{Xi} - 1) \quad (= 0 \iff$ i neuroni sono binari)
- $\frac{B}{2}\Big[\sum_X \sum_i 2^{-i} V_{Xi} - 1\Big]^2 \quad (= 0 \iff$ vale la (2.58)).

A questo punto non resta che aggiungere il termine relativo alla minimizzazione della lunghezza media

$$\frac{A}{2}\Big[\sum_X \sum_i p_X i V_{Xi} - H(S)\Big]^2 \tag{2.63}$$

per ottenere, con semplici passaggi [24],

$$\begin{aligned} T_{XiYj} &= \delta_{XY}[\delta_{ij}(C - F) - C] - p_X p_Y ij - B2^{-(i+j)} - D \\ I_{Xi} &= H(S)p_X i + B2^{-i} + KD + \frac{F}{2} \quad . \end{aligned} \tag{2.64}$$

2.4.5 Il problema del fusto

Veniamo ora al problema su cui abbiamo concentrato la nostra attenzione: si supponga che siano stati assegnati n numeri interi positivi $a_1, \ldots, a_n$ e un intero N che sia la somma di alcuni degli a_i. Il problema che ci si pone è il seguente: dati N e $\mathbf{A} = (a_1, \ldots, a_n)$, trovare $\mathbf{M} = (m_1, \ldots, m_n) \in \{0, 1\}^n$ tale che $N = \mathbf{A}\mathbf{M}^\top$.

Per capire cosa c'entri il fusto (o lo zaino), si pensi all'immagine di un contenitore cilindrico di altezza N (p. e. cm) e si supponga che gli a_i rappresentino le altezze di oggetti cilindrici con alcuni dei quali si deve riempire il contenitore; sarà $m_i = 1$ se decidiamo di mettere a_i nel contenitore, $m_i = 0$ se decidiamo di non metterlo. Naturalmente potrebbero esserci piú soluzioni; in questo caso basta trovarne una qualunque.

Un algoritmo banale per risolvere il problema del fusto è quello della ricerca esauriente: si provano tutte le n-uple binarie finché non se ne trova una che va bene; questo algoritmo, però, è esponenziale, dal momento che le n-uple sono in tutto 2^n; potrebbe darsi che l'n-upla che si cerca fosse proprio l'ultima, rendendo necessari proprio 2^n passi.

Esistono algoritmi piú veloci di questo ma, finora, anche l'algoritmo ottimo, che consiste in circa $2^{\frac{n}{2}}$ passi, risulta esponenziale. Esistono però casi particolari in cui si può risolvere il problema del fusto con un algoritmo polinomiale; sono tutti i casi in cui l'n-upla è *supercrescente* ovvero verifica la condizione

$$a_i > \sum_{j=1}^{i-1} a_j \quad \forall\, i = 2, \ldots, n \quad . \tag{2.65}$$

È facile vedere che in questo caso non è necessario provare tutte le n-uple. Se, per esempio, $N = 154$ e $a_n = 100$, allora deve essere $m_n = 1$ perché la somma dei primi $n-1$ termini è, al massimo, 99; in modo altrettanto semplice si trova il valore di tutti gli altri m_i; in generale, per un'n-upla supercrescente, basta applicare l'algoritmo di Figura 2.6.

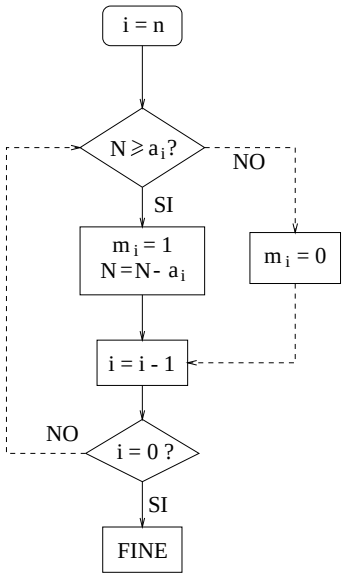


Figura 2.6: Algoritmo facile (N, A).

Osservazione 2.4. Si noti che se $A = (1, 2, 4, 8, \ldots, 2^n)$ il problema del fusto coincide con un convertitore A/D a n bit e il minimo assoluto della funzione energia corrisponde alla scrittura binaria di N. In questo caso può anche essere $N \notin \mathbf{N}$; se cosí è, la rete fornisce la scrittura binaria del numero naturale che meglio approssima N (sempre che si fermi nel minimo assoluto).

2.4.6 Il cifrario del fusto

Il sistema di cifra che va sotto questo nome è dovuto a Ralph Merkle e Mrtin Hellman [25]; esso è basato sulla nozione di *funzione unidirezionale*: si tratta di funzioni invertibili tali che il calcolo della funzione diretta sia facile mentre quello della funzione inversa sia difficile. Tuttavia, anche la funzione inversa risulta facile se si dispone di una particolare informazione detta *molla* che nel cifrario del fusto rappresenta la chiave segreta di decifrazione che consentirà di trasformare un fusto difficile in uno supercrescente. Il messaggio in chiaro del cifrario del fusto è un blocco binario di lunghezza costante mentre il crittogramma è un numero intero.
Si consideri un'n-upla supercrescente $\mathbf{A} = (a_1, \ldots, a_n)$ e si scelga un intero $u > 2a_n$. Di tutti questi valori, viene reso pubblico solo n. Si scelga ancora un intero v minore di u e primo con u. Si calcoli, infine, l'inverso di v nell'aritmentica circolare modulo u: tale inverso è definito come l'unico intero w tale che $vw = 1 \pmod{u}$ e $1 \leqslant w \leqslant u - 1$; nell'ipotesi che v e u siano primi tra loro, l'esistenza e l'unicità di w sono assicurate da un teorema di teoria dei numeri. La chiave di decifrazione è costituita da $\mathbf{A}$, u e w.
Per costruire il crittogramma, si opera nel seguente modo: la sequenza supercrescente viene trasformata nella sequenza difficile $\mathbf{D}$ secondo la formula

$$d_i = va_i \pmod{u} \quad \forall\, i = 1, \ldots, n \tag{2.66}$$

ovvero

$$\mathbf{D} = v\mathbf{A} \pmod{u} \ . \tag{2.67}$$

A questo punto, dato il messaggio in chiaro $\mathbf{M} = (m_1, \ldots, m_n)$, il crittogramma corrispondente è il numero

$$c = \mathbf{D}\mathbf{M}^\top \ . \tag{2.68}$$

Per calcolare $\mathbf{M}$, una spia che intercetti il crittogramma c dovrebbe risolvere il problema del fusto (2.68) che è intrattabile. Il destinatario reale del messaggio, invece, può calcolare

$$c^* = wc \pmod{u} \ ; \tag{2.69}$$

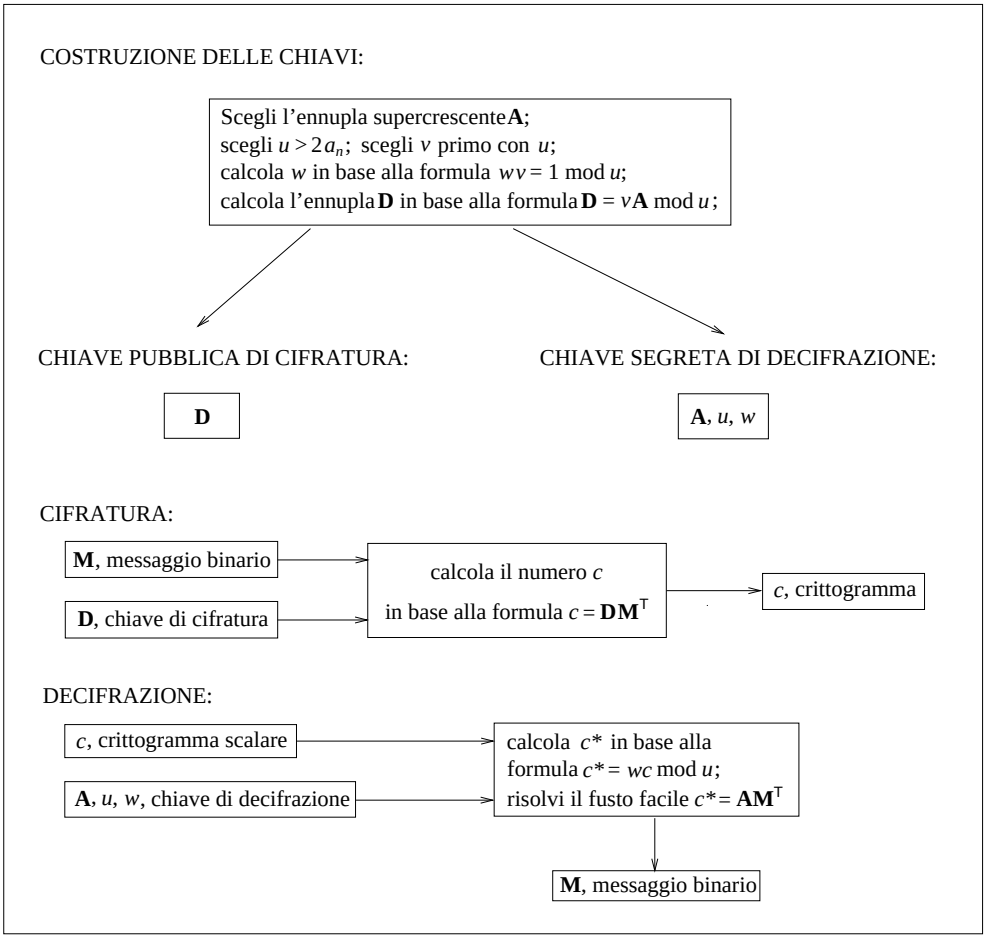


Figura 2.7: Il cifrario del fusto (da [26]).

ricordando la (2.68) e la (2.67), la (2.69) si può scrivere come

$$c^* = (vw)(\mathbf{A}\mathbf{M}^\top) \pmod{u} \tag{2.70}$$

ed essendo $wv = 1 \pmod{u}$, $c^* = \mathbf{A}\mathbf{M}^\top \pmod{u}$. Ma u era stato scelto maggiore di $2a_n$ e quindi, per la supercrescenza di $\mathbf{A}$, è anche maggiore di $a_1 + \cdots + a_n$ e, a maggior ragione, di $\mathbf{A}\mathbf{M}^\top$ che è la somma di alcuni soltanto degli a_i. Allora l'indicazione $\pmod{u}$ è superflua e si può scrivere

$$c^* = \mathbf{A}\mathbf{M}^\top \tag{2.71}$$

che è un problema facile (per una visione schematica riassuntiva si veda la Figura 2.7).
Facciamo ora un esempio nel quale, per ragioni di semplicità, siamo costretti a usare valori piccoli. Sia $\mathbf{A} = (2, 5, 10, 19, 40)$, $u = 85$ e $v = 11$. Poiché $11 \times 31 = 1 \pmod{85}$, $w = 31$; la chiave di decifrazione è $\mathbf{A} = (2, 5, 10, 19, 40)$, $u = 85$, $w = 31$. La chiave pubblica di cifratura è $\mathbf{D} = (22, 55, 25, 39, 15)$: infatti $22 = 11 \times 2 \pmod{85}$, $55 = 11 \times 5 \pmod{85}$, $25 = 11 \times 10 \pmod{85}$, $39 = 11 \times 19 \pmod{85}$ e infine $15 = 11 \times 40 \pmod{85}$. Se il blocco da cifrare è $\mathbf{M} = (0, 0, 1, 0, 1)$, il crittogramma risultante è $c = \mathbf{DM}^\top = 25 + 15 = 40$. Passiamo alla decifrazione: $c^* = 31 \times c \pmod{85}$ ossia $c^* = 50$. Si tratta di risolvere il fusto $50 = \mathbf{AM}^\top$ e l'algoritmo di Figura 2.6 dà subito $\mathbf{M} = (0, 0, 1, 0, 1)$.

2.4.7 Risoluzione del problema del fusto con l'uso di una RN di Hopfield

Data l'altezza N del fusto e l'n-upla $\mathbf{A} = (a_1, \ldots, a_n)$ che rappresenta le altezze di n oggetti, si deve cercare di riempire il fusto nel modo migliore, cioè, in termini matematici, si deve trovare un'n-upla $\mathbf{V} = (V_1, \ldots, V_n)$ tale che $\mathbf{AV}^\top$ sia quanto piú possibile prossimo all'altezza N del fusto.
Si tratta cioè di minimizzare la seguente funzione costo:

$$E = \frac{1}{2}\Big(N - \sum_{i=1}^{n} a_i V_i\Big)^2 = \frac{1}{2}\Big[N^2 + \sum_{i=1}^{n}\sum_{j=1}^{n} a_i a_j V_i V_j - 2N \sum_{i-1}^{n} a_i V_i\Big] \ . \quad (2.72)$$

Aggiungendo il termine necessario per annullare la diagonale (i termini del tipo $a_i a_j V_i V_j$) si ottiene

$$\begin{aligned} E &= \frac{1}{2}\Big(N - \sum_{i=1}^{n} a_i V_i\Big)^2 - \frac{1}{2}\sum_{i=1}^{n} a_i^2 \Big[V_i\big(V_i - 1\big)\Big] = \\ &= -\frac{1}{2}\sum_{i=1}^{n}\sum_{\substack{j=1 \\ j\neq i}}^{n}\Big(-a_i a_j\Big)V_i V_j - \sum_{i=1}^{n}\Big(-\frac{a_i^2}{2} + N a_i\Big)V_i + \frac{N^2}{2} \ . \quad (2.73) \end{aligned}$$

Come nel caso dei codici di Huffman, si ha un termine costante che non influisce sull'evoluzione della rete mentre dai termini restanti si ricava

$$T_{ij} = \begin{cases} -a_i a_j & \text{se } i \neq j \\ 0 & \text{se } i = j \end{cases} \qquad \text{e} \qquad I_i = -\frac{a_i^2}{2} + N a_i \quad . \tag{2.74}$$

Sulla base di quest'ultimo risultato saranno costruite le matrici $\mathbf{T}$ e $\mathbf{I}$ utilizzate nelle simulazioni del quinto capitolo.

Oltre alle difficoltà descritte nel paragrafo 2.4.2, il problema che limita maggiormente l'efficacia dell'impiego dei sistemi neuromorfici in campo combinatorio è che essi non sempre forniscono la soluzione ottima; infatti spesso la funzione di Lyapunov ha molti minimi locali e quando il sistema ne raggiunge uno qualunque si stabilizza. Nel prossimo capitolo, come si diceva, ci si occuperà dei principali metodi per evitare tali minimi.

Capitolo 3

Evitamento dei minimi locali

Come abbiamo visto, nelle RN di Hopfield l'energia definita dalla (2.5) diminuisce nel tempo e su questa proprietà si sviluppa l'idea di utilizzarle per rendere minime le funzioni di costo associate alle soluzioni dei problemi di OC.
La decrescenza della funzione energia, però, costituisce al tempo stesso uno dei limiti dell'applicazione dei sistemi neuromorfici all'OC in quanto essi forniscono soltanto la soluzione piú vicina allo stato iniziale e non sempre questa coincide con la soluzione ottima; la probabilità che ciò accada, infatti, dipende dalla forma della funzione di Lyapunov ovvero dal numero di minimi che essa presenta e dal volume n-dimensionale del dominio di attrazione di ciascun minimo. I metodi proposti, finora, per ovviare a questo inconveniente si possono dividere sommariamente in tre categorie.
La prima prevede la scelta di una forma appropriata della funzione energia, o dei valori dei parametri in essa presenti, in modo da migliorare le proprietà di convergenza della rete avvicinandosi quanto piú possibile alla situazione ideale di una funzione E con un solo minimo. La realizzazione di questa idea, però, ha permesso l'eliminazione di alcuni minimi locali soltanto in alcune reti particolari (vedi paragrafo 3.1) perché nella maggior parte dei casi o si ha una buona conoscenza delle caratteristiche di E - e ciò può comportare un carico computazionale pari alla risoluzione algoritmica del problema - oppure la procedura utilizzata per eliminare un particolare minimo locale ne introduce un altro che nella funzione originaria non c'era. Ad esempio,

nella funzione disegnata col tratto continuo in Figura 3.1(a) si può eliminare il minimo a energia maggiore sommandole la funzione tratteggiata in basso che non modifica né la posizione né il valore di energia del minimo assoluto; si ottiene cosí la funzione di Figura 3.1(b). Ma se non si calcola la giusta

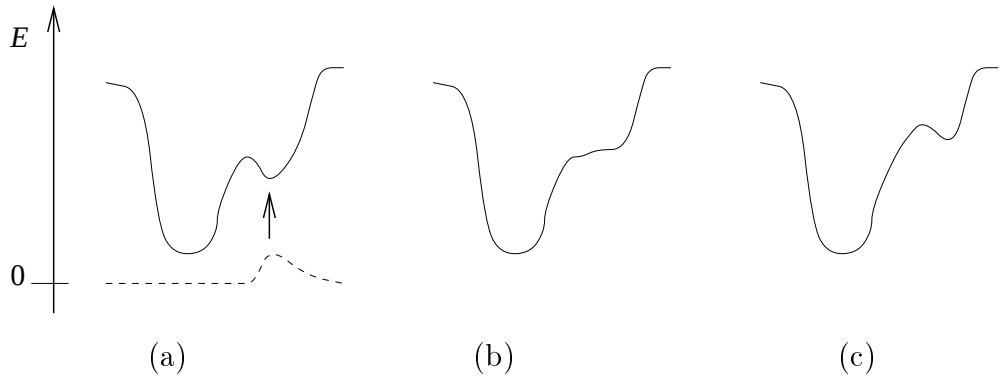


Figura 3.1: Effetto delle modifiche alla funzione energia.

altezza della 'gobba' da aggiungere, si introduce un altro minimo come si vede in Figura 3.1(c).
Un'altra possibilità è quella di introdurre una sorta di rumore termico che genera momentanee transizioni verso stati a energia maggiore permettendo il superamento dei massimi relativi e il raggiungimento di stati stabili con energia minore. Tra i metodi di questo tipo, il piú noto è il *simulated annealing* o *ricottura simulata* (vedi paragrafo 3.3), proposto in origine per i sistemi termodinamici e applicato successivamente a un modello statistico di RN detto *macchina di Boltzmann* (o MB, vedi paragrafo 3.2).
Infine si può alternare l'evoluzione della rete con un algoritmo di evitamento: ogni volta che l'evoluzione porta la rete in un minimo locale, si applica un algoritmo che permette di trovare un punto con la stessa energia (o con energia piú bassa) ma appartenente al dominio di attrazione di un minimo a energia minore. A questo gruppo di metodi appartengono, ad esempio, l'algoritmo del *traforo* (in inglese *tunnel*) e l'algoritmo del rilassamento geometrico proposto nel prossimo capitolo.

3.1 Modifiche della funzione energia

In alcuni articoli apparsi negli ultimi anni ([27],[28]) sono studiate le capacità risolutive delle RN di Hopfield applicate al TSP e in particolare si cerca di far coincidere gli stati stabili con soluzioni accettabili del problema, cioè con percorsi che tocchino tutte le città una e una sola volta. Per questo scopo lo stesso Hopfield aveva introdotto i primi tre termini della (2.34) ma non sempre essi sono sufficienti e capita che la rete si fermi su punti per i quali il quarto termine della sommatoria è cosí piccolo da rendere minimo il livello di energia anche se i primi tre termini non sono nulli.

3.1.1 Considerazioni sulle componenti dell'energia

Nel 1993, Shigeo Abe ha dimostrato che se i parametri A, B, C e D verificano certe ipotesi, allora tutti gli stati stabili sono soluzioni accettabili. Senza entrare nei dettagli della dimostrazione, per i quali si rimanda a [27], si riportano le principali conclusioni:

- se $\mathbf{s}$ è una soluzione dell'equazione $\mathbf{Ts} + \mathbf{I} = 0$ e vale $0 < s_i < 1 \ \forall i = 1, \ldots, n$ allora $\mathbf{s}$ è un punto di equilibrio instabile. Fatto salvo questo risultato, si può congetturare l'impossibilità che la rete finisca in un punto di equilibrio instabile e ci si può limitare a considerare i vertici dell'ipercubo ricavando i seguenti risultati:

- detta $\mathbf{c} = (c_1, \ldots, c_n)$ una generica soluzione accettabile, se le costanti A e B sono positive e se C e D verificano la condizione

$$C > 2D \max_{X,Y,Z}(d_{XY} + d_{XZ}) \quad , \tag{3.1}$$

  allora non ci possono essere stati spuri adiacenti a $\mathbf{c}$ (dove per *adiacente* si intende uno stato ottenuto da $\mathbf{c}$ cambiando una sola delle coordinate c_i da zero a uno o da uno a zero).

- se $A > 0$, $B > 0$ e se vale

$$C > 2D \max_{X} \sum_{Y}^{(4)} d_{XY} \quad , \tag{3.2}$$

dove la sommatoria è estesa alle quattro massime distanze che hanno un indice uguale a X, allora tutti gli stati stabili sono soluzioni accettabili.

3.1.2 L'importanza degli autovalori di T

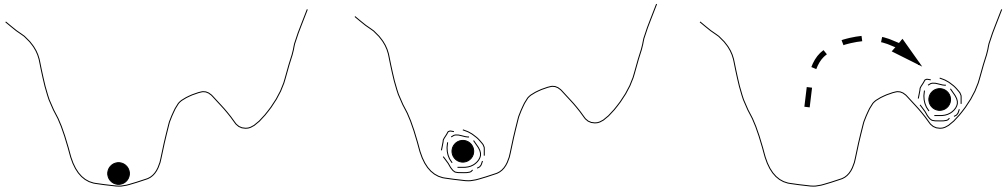
Partendo da un approccio diverso, e cioè ricavando gli autovalori di $\mathbf{T}$ e analizzando la loro influenza sulla dinamica della rete, Sreeram Aiyer, Mahesan Niranian e Frank Fallside [28] dimostrarono che per avere soltanto soluzioni accettabili è sufficiente porre $\mathbf{I} = nC(1, \ldots, 1)^\top$ e modificare gli elementi di T come segue:

$$\begin{aligned} T_{Xi,Yj} &= -A\delta_{XY}(1-\delta_{ij}) - B\delta_{ij}(1-\delta_{XY}) - 2A_1\delta XY\delta_{ij} - \\ &\quad -C + \frac{2(An - A + A_1)}{n^2} - Dd_{XY}(\delta_{j,i+1} + \delta_{j,i-1}) \quad . \end{aligned} \tag{3.3}$$

3.2 La macchina di Boltzmann

Il termine si riferisce a un modello di RN di Hopfield proposto da David Ackley, Geoffrey Hinton e Terence Sejnowski [30]; in esso i neuroni sono binari come quelli di McCulloch e Pitts ma hanno una diversa funzione di trasferimento: il cambiamento di stato della singola unità non è piú una funzione deterministica della somma pesata degli ingressi ma diventa una distribuzione di probabilità sui possibili salti energetici della rete.
Per capire meglio questo concetto, si consideri un sistema fisico molto semplice: una molecola di ossigeno nell'atmosfera; il suo punto a energia minima è il suolo eppure è possibile respirare anche in cima a una montagna poiché l'ossigeno non giace immobile al livello del mare ma è distribuito su tutta l'atmosfera. Questa distribuzione è dovuta all'influenza che la temperatura esercita sul comportamento dei sistemi fisici: nel caso dell'ossigeno, per ciascuna molecola è possibile definire un'energia termica k_BT, proporzionale alla temperatura T della molecola secondo una costante k_B detta *costante di Boltzmann*.
L'energia termica è indicativa dello stato di agitazione della molecola: essa si trova in quiete assoluta, cioè rimane per un tempo indefinito nello stato di equilibrio $\mathbf{V}^e$ in cui si trova, soltanto se $T = 0$ (Figura 3.2(a)); se invece

$T \neq 0$ la molecola si muove in un intorno di $\mathbf{V}^e$ (Figura 3.2(b)). Quanto piú la temperatura è elevata, tanto piú ampie sono le oscillazioni intorno al punto di equilibrio e se T supera un certo valore sono possibili oscillazioni che portano la molecola in uno stato instabile appartenente al dominio di attrazione di un punto di equilibrio diverso da $\mathbf{V}^e$ (Figura 3.2(c)).



(a) (b) (c)

Figura 3.2: Superamento delle colline di potenziale grazie all'agitazione termica.

Si immagini ora che la rete si trovi nello stato $\mathbf{V}_1$ con energia $E(\mathbf{V}_1)$. Sia $\mathbf{V}_2$ lo stato che si ottiene da $\mathbf{V}_1$ cambiando solamente l'uscita di V_{1k}, da zero a uno o da uno a zero, e sia $E(\mathbf{V}_2)$ la sua energia. Nella macchina di Boltzmann la probabilità che l'uscita di V_k cambi, ovvero che la rete si muova da $\mathbf{V}_1$ a $\mathbf{V}_2$, è definita da

$$P = \frac{1}{1 + e^{-\frac{\Delta E_k}{T}}} \qquad \text{dove} \qquad \Delta E_k = E(\mathbf{V}_1) - E(\mathbf{V}_2) \quad . \tag{3.4}$$

In questo modo, se $E(\mathbf{V}_1) < E(\mathbf{V}_2)$, $P < \frac{1}{2}$ cioè è piú probabile che la rete rimanga nello stato $\mathbf{V}_1$; al contrario, se $E(\mathbf{V}_1) > E(\mathbf{V}_2)$, $P > \frac{1}{2}$, cioè è piú probabile che la rete passi nello stato $\mathbf{V}_2$. In entrambi i casi, vengono privilegiati i cambiamenti di stato che fanno diminuire l'energia ma sono sempre possibili anche gli spostamenti verso stati a energia maggiore, che permettono alla rete di superare le colline di potenziale e finire in buche a energia minore.

La (3.4) è la stessa legge che regola la variazione di stato di una particella con due possibili stati di energia; un sistema composto da particelle di questo tipo obbedisce alla statistica di Boltzmann e la probabilità relativa di due stati α e β è:

$$\frac{P_\alpha}{P_\beta} = \frac{e^{-\frac{E_\alpha}{T}}}{e^{-\frac{E_\beta}{T}}} = e^{-\frac{E_\alpha - E_\beta}{T}} \quad . \tag{3.5}$$

Dalla (3.5) si capisce l'influenza che esercita la temperatura sulla probabilità di cambiamento di stato: supponendo $E_\alpha > E_\beta$, se T è prossima allo zero la probabilità che il sistema sia al livello energetico E_α è quasi nulla (Figura 3.3(a)); aumentando T, aumenta corrispondentemente la probabilità di trovare il sistema nello stato α (Figura 3.3(b)), fino a quando, per T molto grande, gli stati α e β diventano equiprobabili (Figura 3.3(c)). In ogni caso, essendo $T > 0$, è sempre $\Pr\{\mathbf{V} = \alpha\} < \Pr\{\mathbf{V} = \beta\}$ a conferma del fatto che un sistema tende spontaneamente a portarsi nello stato a energia minima.

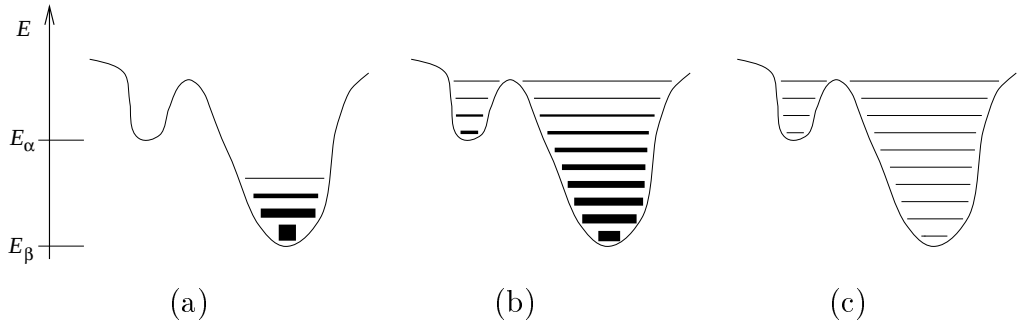


Figura 3.3: Probabilità di variazione di stato a diverse temperature.

3.2.1 Modello statistico e modello deterministico

La (3.4) ha validità generale, indipendentemente dai parametri che definiscono l'energia, me se E ha l'espressione indicata in (2.5), si può esplicitare ΔE_k trovando una relazione tra la macchina di Boltzmann e gli elementi di $\mathbf{T}$:

$$\begin{aligned}
\Delta E_k &= E(\mathbf{V}_1) - E(\mathbf{V}_2) = -\frac{1}{2}\sum_{i,j} T_{ij} V_{1i} V_{1j} + \frac{1}{2}\sum_{i,j} T_{ij} V_{2i} V_{2j} = \\
&= -\frac{1}{2}\sum_{\substack{i\neq k\\ j\neq k}} T_{ij} V_{1i} V_{1j} - \frac{1}{2}\sum_{i} T_{ik} V_{1i} V_{1k} - \frac{1}{2}\sum_{j} T_{kj} V_{1k} V_{1j} + \\
&\quad + \frac{1}{2}\sum_{\substack{i\neq k\\ j\neq k}} T_{ij} V_{2i} V_{2j} + \frac{1}{2}\sum_{i} T_{ik} V_{2i} V_{2k} + \frac{1}{2}\sum_{j} T_{kj} V_{2k} V_{2j} \quad ; \quad (3.6)
\end{aligned}$$

ma $V_{1i} = V_{2i} \ \forall i \neq k$, perciò, posto $\Delta V_k = V_{1k} - V_{2k}$, si ha

$$\Delta E_k = -\frac{1}{2}\sum_i T_{ik} V_{1i} \Delta V_k - \frac{1}{2}\sum_j T_{kj} V_{1j} \Delta V_k \tag{3.7}$$

e per la simmetria di $\mathbf{T}$

$$\Delta E_k = -\Delta V_k \sum_i T_{ik} V_{1i} \quad . \tag{3.8}$$

Sostituendo la (3.8) nella (3.4) si trova

$$P = \frac{1}{1 + e^{\frac{\Delta V_k}{T} \sum_i T_{ik} V_{1i}}} \quad . \tag{3.9}$$

Se $V_{1k} = 0$ e $V_{2k} = 1$, la probabilità che il sistema cambi stato è anche la probabilità che dopo il passaggio di stato l'uscita del k-esimo neurone sia 1 ed è

$$P = \Pr\{V_{2k} = 1\} = \frac{1}{1 + e^{-\frac{1}{T}\sum_i T_{ik} V_{1i}}} \quad . \tag{3.10}$$

Si supponga ora che ciascun neurone di una rete di Hopfield continua sia in realtà un insieme molto numeroso di unità binarie delle quali si osserva solo il comportamento macroscopico. In altre parole, si immagini che l'uscita continua V_k del neurone k-esimo rappresenti la media delle uscite delle unità di cui è composto.
Con queste ipotesi, V_k corrisponde anche alla probabilità empirica che una particolare unità binaria, scelta a caso dall'insieme k-esimo, abbia uscita 1. Poiché per un neurone binario il valore di attivazione u_k è semplicemente la somma algebrica degli ingressi, $u_k = \sum_i T_{ik} V_k$, dalla (1.3) si ha

$$\Pr\left\{V_{2k} = 1\right\} = \frac{1}{1 + e^{-\alpha u_k}} = \frac{1}{1 + e^{-\alpha \sum_i T_{ik} V_{1k}}} \quad , \tag{3.11}$$

dove per semplicità si è supposto $\theta_k = 0$. Confrontando la (3.11) con la (3.10) si può vedere che nel determinare la velocità del passaggio di stato la temperatura T ha nel modello statistico lo stesso peso che ha il guadagno α nel modello deterministico.

3.3 La ricottura simulata

Nel paragrafo precedente è stata messa in luce l'importanza della temperatura nella determinazione del comportamento della rete: alle basse temperature c'è una forte selettività, cioè i livelli energetici di massimi e minimi sono molto diversi e la rete raggiunge in poco tempo il minimo piú vicino allo stato iniziale; se invece la temperatura assume valori elevati, il tempo di risposta diventa piú lungo ma lo stato finale è piú vicino al livello di energia minimo. Un buon espediente per sfruttare gli effetti favorevoli di ciascuna di queste due opposte situazioni è quello di far cominciare l'evoluzione del sistema a una temperatura T_i molto elevata e di raffreddarlo gradatamente. Su questa idea si basa la ricottura simulata (RS) proposta nel 1983 da Kirkpatrick, Gelatt e Vecchi, a proposito dei sistemi termodinamici reali.
Come si può notare in Figura 3.4, col passare del tempo la rete diventa via

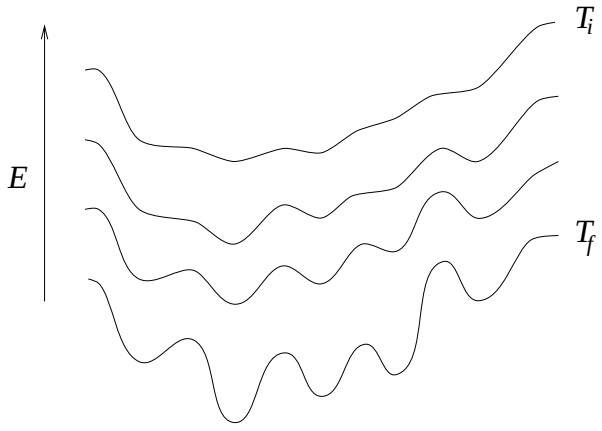


Figura 3.4: Approfondimento dei minimi energetici con la diminuzione di T.

via piú selettiva e diminuisce la probabilità che si verifichi un passaggio di stato che aumenta l'energia. Detto altrimenti, all'inizio la sigmoide (3.4) è molto 'allungata' e al diminuire della temperatura essa si avvicina alla funzione gradino (vedi Figura 3.5); quando la temperatura raggiunge il valore $T = T_f \simeq 0$ sono possibili solo diminuzioni di energia.
In questo modo, all'inizio la rete raggiungerà uno dei punti di minimo relativi

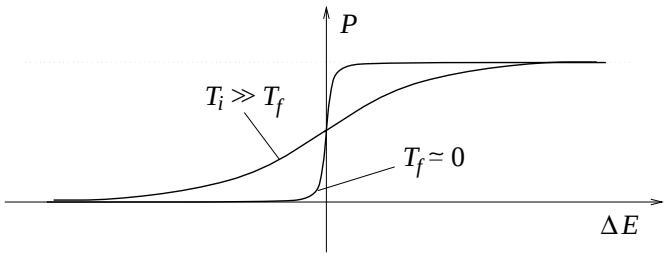


Figura 3.5: Variazione della probabilità di passaggio di stato con la temperatura.

alla grossolana topologia iniziale della funzione energia che a causa dell'elevato valore di T_i non sarà ancora molto ben definita. A mano a mano che la temperatura si abbassa, la rete comincia a rispondere a differenze di energia sempre piú piccole all'interno della regione di attrazione del minimo trovato inizialmente e fornirà una soluzione sempre piú raffinata. Se la regione di attrazione in cui la rete cade inizialmente si trasforma, col passare del tempo, nel dominio relativo al minimo assoluto, la soluzione fornita dal sistema sarà quella ottima.

Osservazione 3.1. La velocità con cui la temperatura viene diminuita durante l'evoluzione della rete rappresenta un parametro determinante per il comportamento del sistema. Se si diminuisce T troppo velocemente, l'energia passa rapidamente da una superficie livellata ad una con valli profonde e si corre il rischio che la rete rimanga bloccata quasi subito in un minimo locale. Al contrario, se T è abbassata troppo lentamente, la superficie dell'energia rimane livellata a lungo e quindi ed è molto probabile che la rete esca, oltre che dai minimi locali, anche da quello globale. Trovare la procedura ottima rimane il punto cruciale nella realizzazione della RS.

3.4 Modifica temporanea delle connessioni

Ispirati dalla RS, Mengkang Peng, Narendra Gupta e Alistair Armitage [32] proposero un'idea innovativa per sfuggire dai minimi locali e la realizzarono

in un algoritmo che chiamarono Local Minima Escape (LME). Esso è una combinazione di tecniche di disturbo termico e delle proprietà di deriva delle reti di Hopfield verso un punto di equilibrio.
Per capire come funziona l'algoritmo, si immagini di variare i parametri della rete H, cioè le connessioni T_{ij} e le polarizzazioni I_i, sommandovi un rumore gaussiano:

$$T'_{ij} = (1 + \alpha^T r^T_{ij}) T_{ij} + \beta^T r^T_{ij} \quad (3.12)$$

$$I'_i = (1 + \alpha^I r^T_{ij}) I_i + \beta^I r^I_i \quad , \quad (3.13)$$

dove r^T_{ij} e r^I_i sono rumori gaussiani standard (con $r^T_{ji} = r^T_{ij}$) e α^T, β^T, α^I e β^I sono costanti positive che controllano l'intensià del disturbo.
In questo modo si ottiene una nuova rete H' i cui neuroni possono essere messi in relazione biunivoca naturale con quelli di H. Inoltre gli stati di una delle due reti possono essere trasformati in stati dell'altra. Si supponga ora che H si trovi in un minimo locale $\mathbf{V}^e$; si applichino le (3.12) e (3.13) creando H' e si assegni all'uscita di ciascun neurone p' di H' lo stesso valore di uscita del corrispondente neurone p di H. Se $\mathbf{V}^e$ non è uno stato di equilibrio per H', quest'ultima evolverà e convergerà verso un suo stato di equilibrio $\mathbf{W}^e$ che avrà un'energia $E'(\mathbf{W}^e)$ minore di $E'(\mathbf{V}^e)$. Il passo successivo è quello di assegnare $\mathbf{W}^e$ come stato di partenza di H per una seconda evoluzione. Se partendo da $\mathbf{W}^e$ H raggiunge un nuovo stato di equilibrio, questo viene accettato al posto di $\mathbf{V}^e$, altrimenti si mantiene $\mathbf{V}^e$; con questa assegnazione si completa un'iterazione.
L'iterazione successiva compirà lo stesso processo cominciando dalla generazione di due nuovi rumori. L'algoritmo termina quando è stato effettuato un numero predefinito N_i di iterazioni o quando dopo un numero N_t di tentativi non ha permesso di trovare un nuovo minimo.
Il metodo differisce dalla RS principalmente in due punti:

- La RS prevede sempre la possibilità che l'energia aumenti, e la stabilità del sistema, ottenuta con il raffreddamento, non garantisce la convergenza verso il minimo globale in un tempo finito. Nell'algoritmo LME, invece, la rete si sposta sempre nella direzione della diminuzione di energia e ciò permette al sistema di finire nel minimo globale in un tempo finito.

- Nella RS, prima che la procedura di ricottura sia prossima alla fine lo stato corrente non fornisce risultati significativi per il problema, mentre con il metodo LME ogni volta che termina l'evoluzione di una delle due reti si ha una soluzione significativa.

Una trattazione teorica delle prestazioni dell'algoritmo LME è molto difficile ma le simulazioni fatte dagli autori hanno mostrato che esso, rispetto alla RS, raggiunge soluzioni migliori e in un tempo piú breve. Ma hanno anche messo in luce che quando l'energia scende sotto un certo valore diventa molto difficile diminuirla ulteriormente.

3.5 L'algoritmo del traforo

È un metodo di tipo perturbativo valido per tutti i problemi di minimizzazione ed è stato applicato alle reti neurali per la prima volta nel 1992 da Kwok-wai Cheung e Tong Lee [33]. Concettualmente esso consiste in quanto segue:

- si esegue una normale evoluzione della rete e quando essa finisce in un minimo locale $\bar{\mathbf{v}}$, si modifica la funzione energia, aggiungendole un polo, facendo in modo che $\bar{\mathbf{v}}$ non sia piú punto di minimo. La funzione cosí modificata è detta funzione *di traforo* (in inglese *tunneling*) ed indicata con F_t

- sulla base di F_t, si calcolano due nuove matrici T' e I' e si dà inizio alla fase di traforazione consistente in un'evoluzione della rete definita da T' e I'

- la fase di traforo termina quando si raggiunge un punto $\mathbf{w}$ per il quale $F_t(\mathbf{w}) < 0$

- si torna al primo punto considerando $\mathbf{w}$ come stato iniziale di una nuova evoluzione della rete originaria (si veda la Figura 3.6).

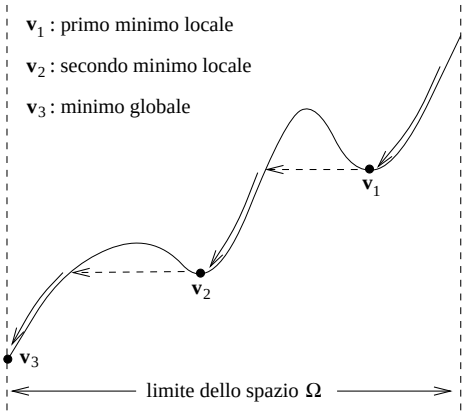


Figura 3.6: Il metodo del traforo.

La funzione oggetto modificata $F_t(\mathbf{v})$ è definita da

$$F_t(\mathbf{v}) = \frac{E(\mathbf{v}) - E(\bar{\mathbf{v}})}{\sum_{i=1}^{n}(V_i - \bar{V}_i)^2} \quad . \tag{3.14}$$

La sua scelta, possibilmente, deve comportare una realizzazione circuitale semplice e per questo motivo gli autori suggeriscono che venga utilizzato solo il polo relativo all'ultimo minimo incontrato. Inoltre, per evitare che il sistema si blocchi su eventuali minimi $\mathbf{x}$ della funzione F_t, essi propongono di traslare il polo corrente in $\mathbf{x}$ in modo che la rete scivoli via da $\mathbf{x}$. Si indichi ora con D_t il denominatore della funzione di traforo.
Calcolando la derivata parziale di F_t lungo la direzione i-esima, si trova

$$\frac{\partial F_t(\mathbf{v})}{\partial V_i} = \frac{D_t \dfrac{\partial E(\mathbf{v})}{\partial V_i} - \left(E(\mathbf{v}) - E(\bar{\mathbf{v}})\right)\dfrac{\partial D_t}{\partial V_i}}{D_t^2} \quad . \tag{3.15}$$

Affinché $\Delta F_t(\mathbf{v})$ sia sempre minore di zero, deve essere $\Delta V_i = -\alpha \dfrac{\partial F_t(\mathbf{v})}{\partial V_i}$, cioè

$$\Delta V_i = \alpha \frac{-D_t \dfrac{\partial E(\bar{\mathbf{v}})}{\partial V_i} + \left(E(\mathbf{v}) - E(\bar{\mathbf{v}})\right)\dfrac{\partial D_t}{\partial V_i}}{D_t^2} \quad . \tag{3.16}$$

Questa quantità è utilizzata come argomento di una funzione gradino g per definire la regola di transizione dello stato durante la fase di traforo:

$$V_i = g\left(\frac{-D_t\dfrac{\partial E(\bar{\mathbf{v}})}{\partial V_i} + \left(E(\mathbf{v}) - E(\bar{\mathbf{v}})\right)\dfrac{\partial D_t}{\partial V_i}}{D_t^2}\right) \tag{3.17}$$

dove

$$E(\mathbf{v}) = \left[\sum_{j=1}^{n} T_{ij}V_iV_j - \sum_{i=1}^{n} I_iV_i\right] \qquad D_t = \sum_{i=1}^{n}(V_i - \bar{V}_i)^2 \tag{3.18}$$

$$\frac{\partial E(\mathbf{v})}{\partial x_i} = -\left[\sum_{j=1}^{n} T_{ij}V_j + I_i\right] \qquad \frac{\partial D_t}{\partial x_i} = 2(x_i - \bar{x}_i) \quad . \tag{3.19}$$

Nelle loro simulazioni, Cheung e Lee notarono che il tempo richiesto per migliorare una soluzione prossima alla soluzione ottima è molto lungo a causa dell'elevato numero di minimi nei quali si incorre durante la fase di traforo. Dunque l'algoritmo risulta veloce quando si è ancora lontani dalla soluzione ottima ma, a mano a mano che ci si avvicina ad essa, il tempo di attesa cresce e l'algoritmo si dimostra inefficiente.

Capitolo 4

Metodi geometrici

4.1 Considerazioni sul cifrario del fusto

Come si è detto nel capitolo 2, per ottenere il crittogramma del cifrario del fusto si opera sul messaggio binario in chiaro $(V_1, \ldots, V_n)$ nel modo seguente: assegnata l'n-upla di numeri interi $\mathbf{a} = (a_1, \ldots, a_n)$, si calcola

$$N = \sum_{i=1}^{n} a_i V_i \quad . \tag{4.1}$$

Decrittare il messaggio, allora, significa trovare l'n-upla $\mathbf{v} = (V_1, \ldots, V_n)$ che verifica la (4.1) noti N e $\mathbf{a}$.
Se si guarda al problema del fusto dal punto di vista esclusivamente crittanalitico, si può vedere che la sua risoluzione è semplificata in quanto la (4.1) è una funzione lineare: date due soluzioni $\mathbf{X} = (X_1, \ldots, X_n)$ e $\mathbf{Y} = (Y_1, \ldots, Y_n)$, tali che $\sum_{i=1}^{n} X_i a_i = N_X$ e $\sum_{i=1}^{n} Y_i a_i = N_Y$, si ha

$$\sum_{i=1}^{n} (X_i + Y_i) a_i = N_X + N_Y \quad . \tag{4.2}$$

Un cifrario lineare è di semplice applicazione ma la linearità ne costituisce al tempo stesso il punto debole. Il cifrario del fusto non fa eccezione; se infatti si fa l'ipotesi, peraltro non molto restrittiva, che non tutti gli a_i siano pari, allora guardando la cifra meno significativa del messaggio cifrato N si ottiene un bit d'informazione sul messaggio in chiaro: se essa è pari, anche il numero

n_d degli oggetti di altezza dispari che vengono messi nel fusto deve essere pari; se la cifra è dispari, anche n_d deve essere dispari.
A questa facile critica se ne possono aggiungere altre, la piú interessante delle quali ha permesso a Shamir di dimostrare che è possibile forzare il cifrario con un algoritmo di complessità $O(n^4)$.
L'obiettivo di questa tesi è quello di valutare la possibilità di risolvere il problema del fusto, indipendentemente dalle sue applicazioni, utilizzando il metodo neurale invece degli algoritmi tradizionali.

4.2 Studio geometrico della funzione energia

Ricordiamo l'espressione dell'energia per le RN di Hopfield:

$$E = -\frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} T_{ij}V_iV_j - \sum_{i=1}^{n} I_iV_i + \frac{N^2}{2} \tag{4.3}$$

dove

$$T_{ij} = \begin{cases} -a_ia_j & \text{se } i \neq j \\ 0 & \text{se } i = j \end{cases} \qquad \text{e} \qquad I_i = -\frac{a_i^2}{2} + Na_i \quad . \tag{2.74}$$

Lo scoglio principale in cui ci si imbatte, come già accennato alla fine del capitolo 2, è rappresentato dai minimi locali, ovvero dai punti dello spazio Ω che corrispondono a soluzioni diverse dalla soluzione ottima pur essendo, magari,

$$\sum_{i=1}^{n} a_iV_i \simeq N \quad . \tag{4.4}$$

La situazione in cui ci si viene a trovare è simile a quella di Figura 4.1 dove, per semplicità, la superficie dell'energia, che in realtà è n-dimensionale, è una curva; in questa semplificazione gli iperpiani a energia costante $E = cost$ sono rette parallele all'asse delle ascisse. Come si è visto nel capitolo precedente, vi sono diversi metodi per evitare i minimi locali o per sbloccare la rete quando l'evoluzione la porti in uno di essi, ma tutti sono validi solo per un particolare problema o per minimi che soddisfino certe caratteristiche. Gli unici metodi di validità generale sono la macchina di Boltzmann e la RS,

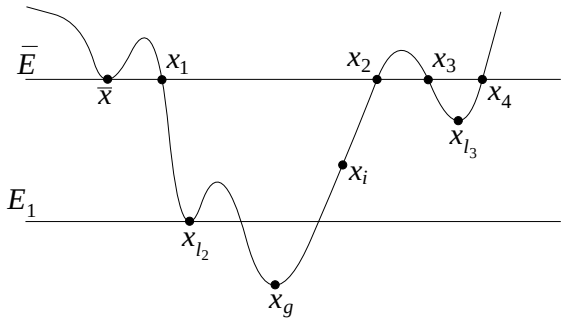


Figura 4.1: Funzione energia e minimi locali.

ma spesso risultano troppo lenti.
In questa tesi si vuole intraprendere una strada diversa, sfruttando le proprietà geometriche della funzione energia per ricavare dei punti corrispondenti a un livello di energia voluto. Si supponga, per esempio, di conoscere un punto x_i appartenente al dominio di attrazione del minimo globale x_g (vedi Figura 4.1); se si impone x_i come stato iniziale della rete, l'evoluzione porterà il sistema proprio in x_g fornendo la soluzione ottima. Su quest'idea si basa il primo algoritmo proposto che è stato chiamato *inizializzazione geometrica* [34][35].
Il secondo algoritmo, detto *rilassamento geometrico* [34][35] prende spunto da un'idea simile: si supponga che dopo una prima evoluzione la rete finisca in un punto $\bar{x}$ con energia $\bar{E}$ (si veda ancora la Figura 4.1); se si riesce a trovare un punto x_1 con la stessa energia $\bar{E}$ (o con energia minore) che non sia a sua volta un punto di minimo e si fa ripartire il sistema da x_1, è chiaro che esso si fermerà in un minimo ad energia $E_1 < \bar{E}$. Ripetendo questo procedimento finché la rete si blocca su altri minimi locali, si raggiunge la soluzione ottima in un numero finito di passi, dal momento che il numero dei minimi locali è finito.
Purtroppo, come si vedrà, anche questi due metodi non sono esenti da debolezze; quella che piú si fa sentire è una notevole difficoltà nella ricerca dei punti desiderati per reti con un numero elevato di neuroni, difficoltà che determina una corrispondente diminuzione di efficacia degli algoritmi.

Ritorniamo, ora, alla funzione energia; la 4.3 può essere scritta in forma matriciale come:

$$E = \frac{1}{2}\mathbf{v}^\top \mathbf{A}\mathbf{v} + \mathbf{b}^\top \mathbf{v} + c \quad , \tag{4.5}$$

dove

$$\mathbf{A} = \begin{pmatrix} 0 & a_1a_2 & \cdots & a_1a_n \\ a_2a_1 & 0 & \cdots & \vdots \\ \vdots & \vdots & \ddots & a_{n-1}a_n \\ a_na_1 & \cdots & a_na_{n-1} & 0 \end{pmatrix}, \quad \mathbf{b} = \begin{pmatrix} \frac{a_i^2}{2} - xa_i \\ \vdots \\ \frac{a_n^2}{2} - xa_n \end{pmatrix} \quad \text{e} \quad c = \frac{N^2}{2} \quad . \tag{4.6}$$

Si operi il cambio di variabili $\mathbf{v} = \mathbf{U}\mathbf{w}$, dove $\mathbf{U}$ è una matrice unitaria, ossia tale che $\mathbf{U}^\mathsf{H}\mathbf{U} = \mathbf{U}\mathbf{U}^\mathsf{H} = \mathrm{I}$ - dove $\mathbf{U}^\mathsf{H}$ è la trasposta coniugata di $\mathbf{U}$ - ovvero, dal momento che trattiamo con numeri reali, tale che $\mathbf{U}^{-1} = \mathbf{U}^\top$. Allora vale:

$$\begin{aligned} E &= \frac{1}{2}\mathbf{v}^\top \mathbf{U}\mathbf{U}^{-1}\mathbf{A}\mathbf{U}\mathbf{U}^{-1}\mathbf{v} + \mathbf{b}^\top \mathbf{U}\mathbf{U}^{-1}\mathbf{v} + c = \\ &= \frac{1}{2}\left(\mathbf{U}^\top \mathbf{v}\right)^\top \mathbf{U}^{-1}\mathbf{A}\mathbf{U}\mathbf{U}^{-1}\mathbf{v} + \mathbf{b}^\top \mathbf{U}\mathbf{U}^{-1}\mathbf{v} + c = \\ &= \frac{1}{2}\mathbf{w}^\top \mathbf{D}\mathbf{w} + \mathbf{d}^\top \mathbf{w} + c \end{aligned} \tag{4.7}$$

dove si è posto $\mathbf{d}^\top = \mathbf{b}^\top \mathbf{U}$ e

$$\mathbf{D} = \mathbf{U}^{-1}\mathbf{A}\mathbf{U} = \begin{pmatrix} \lambda_1 & 0 & \cdots & 0 \\ 0 & \lambda_2 & & \vdots \\ \vdots & & \ddots & 0 \\ 0 & \cdots & 0 & \lambda_n \end{pmatrix} \quad . \tag{4.8}$$

L'esistenza e la forma della matrice $\mathbf{D}$, sono assicurate dalla simmetria di $\mathbf{A}$ che ne permette la diagonalizzazione. La matrice di diagonalizzazzione è $\mathbf{U}$ e gli elementi della diagonale di $\mathbf{D}$ sono gli autovalori di $\mathbf{A}$. Essendo $\mathbf{A}$ simmetrica, questi autovalori sono numeri reali. Da un teorema di algebra lineare, si sa che la somma degli autovalori di una matrice è uguale alla traccia della matrice, ovvero alla somma degli elementi della diagonale. Per

la matrice **A**, dunque, si ha:

$$\sum_{i=1}^{n} \lambda_i = 0 \quad \text{o anche} \quad \sum_{i=1}^{n-1} \lambda_i = \lambda_n \quad . \tag{4.9}$$

Allora si può scrivere

$$\begin{aligned} E &= \frac{1}{2}\sum_{i=1}^{n} \lambda_i w_i^2 + \sum_{i=1}^{n} d_i w_i + c = \\ &= \sum_{i=1}^{n} \frac{\lambda_i}{2}\left(w_i^2 + 2\frac{d_i}{\lambda_i} w_i \right) + c = \\ &= \sum_{i=1}^{n} \frac{\lambda_i}{2}\left(w_i^2 + 2\frac{d_i}{\lambda_i} w_i + \left(\frac{d_i}{\lambda_i}\right)^2 - \left(\frac{d_i}{\lambda_i}\right)^2 \right) + c = \\ &= \sum_{i=1}^{n} \frac{\lambda_i}{2}\left(w_i + \frac{d_i}{\lambda_i} \right)^2 - \sum_{i=1}^{n} \frac{\lambda_i}{2}\left(\frac{d_i}{\lambda_i}\right)^2 + c \quad . \end{aligned} \tag{4.10}$$

Se si pone

$$x_i = w_i + \frac{d_i}{\lambda_i} \quad , \tag{4.11}$$

si ottiene

$$E = \sum_{i=1}^{n} \frac{\lambda_i}{2} x_i^2 - \sum_{i=1}^{n} \frac{\lambda_i}{2}\left(\frac{d_i}{\lambda_i}\right)^2 + c \quad . \tag{4.12}$$

Definizione 4.1. *Una matrice* **F** *è detta* normale *se* $\mathbf{F}\mathbf{F}^{\mathsf{H}} = \mathbf{F}^{\mathsf{H}}\mathbf{F}$.

Definizione 4.2. *Una matrice* **F** *è detta* diagonale *se* $G_{ij} = 0 \quad \forall i \neq j$.

Teorema 4.1. *Gli autovalori di* **A** *soddisfano le diseguaglianze* $\lambda_i < 0 \quad \forall i : 1 \leqslant i \leqslant n-1$ *e* $\lambda_n > 0$.

Dimostrazione. Per la dimostrazione si può partire da un teorema di algebra lineare [36] il quale afferma che se **F** è una matrice normale con autovalori $\mu_1, \ldots, \mu_n$ e **G** è una matrice diagonale reale, allora gli autovalori di $\mathbf{F} + \mathbf{G}$ stanno nel cerchio complesso

$$|z - (\alpha + \mu_j)| \leqslant \beta \tag{4.13}$$

dove, indicato con g_i il generico autovalore di $\mathbf{G}$, $2\alpha = \max_j g_j + \min_j g_j$ e $2\beta = \max_j g_j - \min_j g_j$. Nel nostro caso, se si pone

$$\mathbf{F} = \begin{pmatrix} a_1^2 & a_1 a_2 & \cdots & a_1 a_n \\ a_2 a_1 & a_2^2 & \cdots & a_2 a_n \\ \vdots & \vdots & \ddots & \vdots \\ a_n a_1 & a_n a_2 & \cdots & a_n^2 \end{pmatrix} \quad \text{e} \quad \mathbf{G} = \begin{pmatrix} -a_1^2 & 0 & \cdots & 0 \\ 0 & -a_2 & \cdots & 0 \\ 0 & \cdots & 0 & -a_n^2 \end{pmatrix} \quad , \tag{4.14}$$

si può ricavare, con semplici passaggi, la seguente disequazione per il generico autovalore λ_i di $\mathbf{A} = \mathbf{F} + \mathbf{G}$:

$$\mu_i + \min_i g_i \leqslant \lambda_i \leqslant \mu_i + \max_i g_i \quad . \tag{4.15}$$

Essendo $g_i = -a_i^2$, detti $a = \min_i a_i$ e $A = \max_i a_i$, la (4.15) diventa

$$\mu_i - A^2 \leqslant \lambda_i \leqslant \mu_i - a^2 \quad , \tag{4.16}$$

ma $\mathbf{F}$ ha caratteristica 1 e per i suoi autovalori vale

$$\mu_i = 0 \quad \forall i : 0 \leqslant i \leqslant n-1 \quad \text{e} \quad \mu_n = \sum_{i=1}^{n} a_i^2 \geqslant 0 \quad . \tag{4.17}$$

Sostituendo la (4.17) nella (4.16), si ricava, in particolare,

$$\lambda_i \leqslant -a^2 \leqslant 0 \quad \forall i : 0 \leqslant i \leqslant n-1 \quad \text{e} \quad 0 \leqslant \sum_{I=1}^{n} a_i - A^2 \leqslant \lambda_n \quad . \tag{4.18}$$

□

Dunque si può porre $p_i = -\lambda_i$ per $0 \leqslant i \leqslant n-1$ e $p_n = \lambda_n$. In questo modo dalla (4.12) si ha

$$\sum_{i=1}^{n-1} \frac{p_i}{2} x_i^2 - \frac{p_n}{2} x_n^2 = \sum_{i=1}^{n-1} \frac{p_i}{2} \left(\frac{d_i}{p_i}\right)^2 - \frac{p_n}{2} \left(\frac{d_n}{p_n}\right)^2 + c - E \quad . \tag{4.19}$$

Dato che $p_i > 0$, si possono definire le quantità $\alpha_i = \sqrt{\dfrac{2}{p_i}}$ ottenendo

$$\sum_{i=1}^{n-1} \frac{x_i^2}{\alpha_i^2} - \frac{x_n^2}{\alpha_n^2} = \frac{N^2}{2} - E + \sum_{i=1}^{n-1} \frac{p_i}{2} \left(\frac{d_i}{p_i}\right)^2 - \frac{p_n}{2} \left(\frac{d_n}{p_n}\right)^2 \quad . \tag{4.20}$$

Se, infine, si pone

$$k = -\frac{N^2}{2} + E - \sum_{i=1}^{n-1} \frac{p_i}{2}\left(\frac{d_i}{p_i}\right)^2 + \frac{p_n}{2}\left(\frac{d_n}{p_n}\right)^2 \quad , \tag{4.21}$$

si ricava

$$\frac{x_n^2}{\alpha_n^2} - \sum_{i=1}^{n-1} \frac{x_i^2}{\alpha_i^2} = k \quad . \tag{4.22}$$

Per valori di E costanti, la (4.22) rappresenta l'equazione di un iperboloide n-dimensionale. Se $k > 0$ questo iperboloide ha due falde, e nel caso tridimensionale assume la forma di Figura 4.2. Se si diminuisce il valore di

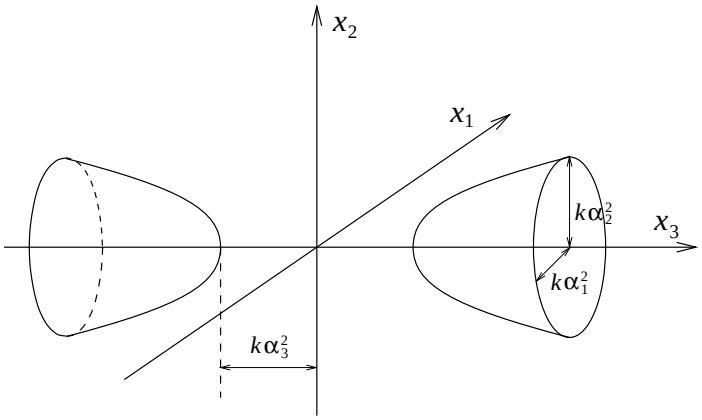


Figura 4.2: Iperboloide a due falde.

E, diminuisce k e decresce anche $k\alpha_n^2$ cioè la distanza delle falde dall'origine del sistema di assi cartesiani. Per E abbastanza piccolo, si ha $k = 0$ e l'iperboloide degenera nei suoi asintoti (vedi Figura 4.3). Per valori di E ancora piú piccoli, k diventa negativo e si ottiene un'iperboloide a una falda (Figura 4.4). Dunque la funzione energia per la rete relativa al problema del fusto assume una forma particolare, che presenta una forte simmetria rispetto all'asse x_n. Con il cambiamento di variabili definito dalla (4.11), lo spazio degli stati che la rete può raggiungere - cioè l'ipercubo $\Omega = \left\{\mathbf{v} = (V_1, \ldots, V_n) : 0 \leqslant V_i \leqslant 1 \ \ \forall i = 1, \ldots, n\right\}$ - è trasformato in un

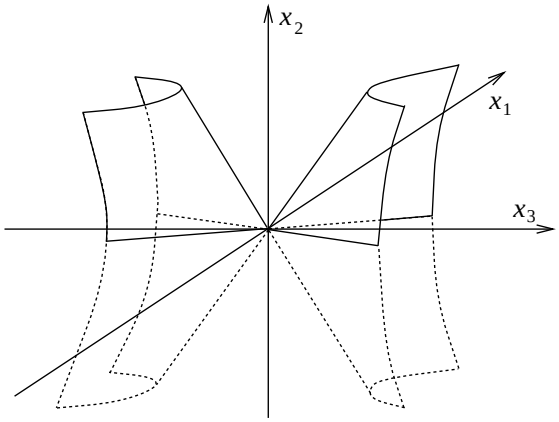


Figura 4.3: Iperboloide degenere coincidente con i suoi asintoti.

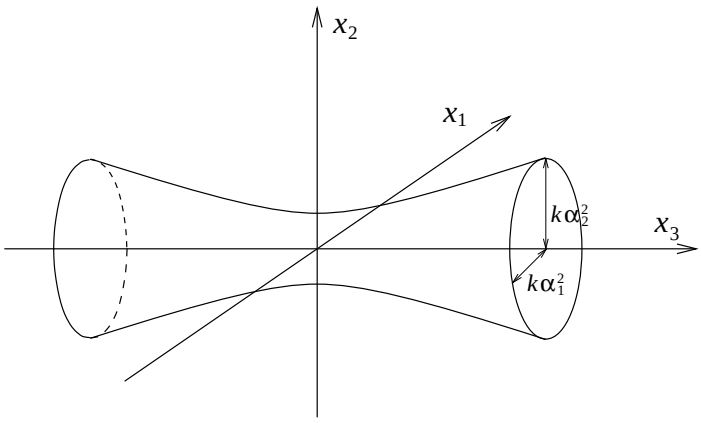


Figura 4.4: Iperboloide a una falda.

parallelepipedo n-dimensionale Ω' a facce piane parallele (vedi Figura 4.5). È facile capire che se un vertice $\bar{\mathbf{x}}$ di questo parallelepipedo corrisponde a un minimo di E, allora l'iperboloide $E = E(\bar{\mathbf{x}}) = \bar{E}$ è tangente a Ω' in $\bar{\mathbf{x}}$. Infatti se cosí non fosse, ovvero se in un intorno di $\bar{\mathbf{x}}$ i punti di Ω' che stanno

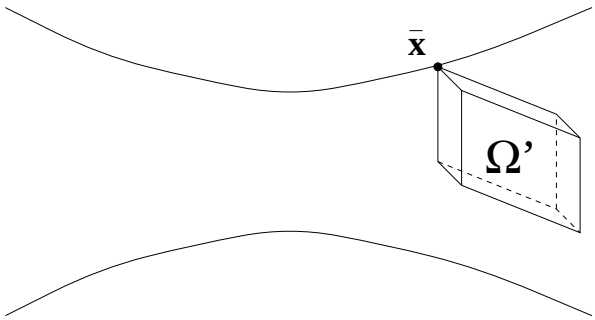


Figura 4.5: Lo spazio Ω' degli stati possibili dopo la trasformazione (4.11).

sull'iperboloide $E = \bar{E}$ fossero piú di uno come in Figura 4.6, nello stesso intorno ci sarebbero punti $\mathbf{z}$ appartenenti a iperboloidi a energia minore di $\bar{E}$ contraddicendo l'ipotesi che $\bar{\mathbf{x}}$ fosse di minimo.

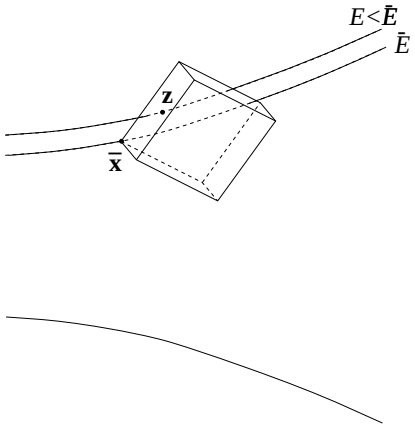


Figura 4.6: Andamento degli iperboloidi nei vertici che non rappresentano punti di minimo.

4.3 Inizializzazione geometrica (IG)

Si tratta di trovare un punto privilegiato dell'ipercubo n-dimensionale da cui far partire l'evoluzione della rete. Il punto ricercato dovrebbe essere abbastanza vicino alla soluzione ottima in modo che la probabilità che appartenga al dominio di attrazione del minimo assoluto sia molto alta.
Se la soluzione al problema del fusto è unica, il vertice $\bar{\mathbf{x}}$ corrispondente a N è l'unico vertice dell'ipercubo appartenente all'iperboloide $E = 0$ (si veda la Figura 4.7 nella quale, per chiarezza, si rappresenta il caso bidimensionale). Dal momento che prima di compiere la trasformazione (4.11) tutti

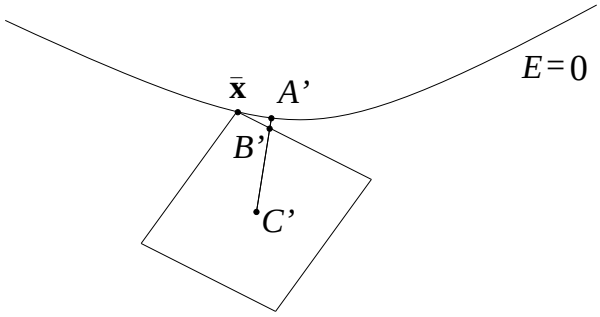


Figura 4.7: Punto a distanza minima, vicino alla soluzione.

i vertici dell'ipercubo hanno la stessa distanza dal centro $C = (\frac{1}{2}, \ldots, \frac{1}{2})$, è ragionevole pensare che C', immagine di C tramite la (4.11), abbia una distanza minima dall'iperboloide $E = 0$ proprio nelle immediate vicinanze di $\bar{\mathbf{x}}$. L'algoritmo è costitiuto dai seguenti passi:

- si calcola il punto A' sull'iperboloide a distanza minima dal centro C' dell'immagine Ω' dell'ipercubo
- si congiunge A' con C' mediante una retta
- si calcola il punto B' di intersezione di questa retta con la superficie dell'ipercubo
- si fa partire l'evoluzione della rete dal punto B immagine di B' tramite la trasformazione inversa della (4.11).

Vediamo ora come calcolare le coordinate del punto A'. L'equazione che definisce l'iperboloide a energia E è:

$$\sum_{i=1}^{n-1} \frac{x_i^2}{\alpha_i^2} - \frac{x_n^2}{\alpha_n^2} = -k \quad , \tag{4.23}$$

oppure, se si pone $\delta = -k + E$,

$$\sum_{i=1}^{n-1} \frac{x_i^2}{\alpha_i^2} - \frac{x_n^2}{\alpha_n^2} = \delta - E \quad , \tag{4.24}$$

in cui compare il valore E dell'energia. Il generico punto P di questo iperboloide ha coordinate

$$P = \left\{ x_i, \ldots, x_{n-1}, x_n = \pm \sqrt{\alpha_n^2 \left(\sum_{i=1}^{n-1} \frac{x_i^2}{\alpha_i^2} - \delta + E \right)} \right\} \quad . \tag{4.25}$$

Il quadrato della distanza di P da C' è:

$$d^2(C', P) = \sum_{i=1}^{n-1} (x_i - x_i^{(c)})^2 + \left(\pm \sqrt{\alpha_n^2 \left(\sum_{i=1}^{n-1} \frac{x_i^2}{\alpha_i^2} - \delta + E \right)} - x_n^{(c)} \right)^2 \tag{4.26}$$

dove $\mathbf{x}^{(c)} = (x_1^{(c)}, \ldots, x_n^{(c)}) = \mathbf{U}^\top (\frac{1}{2}, \ldots, \frac{1}{2})^\top + \mathbf{d}\mathbf{D}^{-1}$. Per trovare i punti P che rendono minima $d(C', P)$ si devono porre uguali a zero le derivate parziali della (4.26):

$$\frac{\partial d^2(C', P)}{\partial x_i} = 2(x_i - x_i^{(c)}) + 2 \left(\pm \sqrt{\alpha_n^2 \left(\sum_{i=1}^{n-1} \frac{x_i^2}{\alpha_i^2} - \delta + E \right)} - x_n^{(c)} \right) \cdot$$

$$\cdot \left(\pm \frac{1}{\sqrt{\alpha_n^2 \left(\sum_{i=1}^{n-1} \frac{x_j^2}{\alpha_j^2} - \delta + E \right)}} \frac{\alpha_n^2}{2} \frac{2x_i}{\alpha_i^2} \right) = 0 \quad . \tag{4.27}$$

Dividendo per 2 ed arrangiando i termini, si ottiene, $\forall i : 1 \leqslant i \leqslant n-1$,

$$x_i\left[1+\left(\frac{\alpha_n}{\alpha_i}\right)^2\right]-x_i^{(c)}=\pm\frac{x_i x_n^{(c)}\left(\frac{\alpha_n}{\alpha_i}\right)^2}{\sqrt{\alpha_n^2\left(\sum_{j=1}^{n-1}\frac{x_j^2}{\alpha_j^2}-\delta+E\right)}} \quad , \tag{4.28}$$

cioè un sistema non lineare di $n-1$ equazioni in $n-1$ incognite. La (4.28), però, può essere riscritta come

$$\sqrt{\alpha_n^2\left(\sum_{j=1}^{n-1}\frac{x_j^2}{\alpha_j^2}-\delta+E\right)}=\pm\frac{x_i x_n^{(c)}\left(\frac{\alpha_n}{\alpha_i}\right)^2}{x_i\left[1+\left(\frac{\alpha_n}{\alpha_i}\right)^2\right]-x_i^{(c)}} \tag{4.29}$$

in cui il termine al primo membro è costante con i. Quindi per ogni coppia $(i,j) : 1 \leqslant i \leqslant n-1, 1 \leqslant j \leqslant n-1$ vale

$$\frac{x_i x_n^{(c)}\left(\frac{\alpha_n}{\alpha_i}\right)^2}{x_i\left[1+\left(\frac{\alpha_n}{\alpha_i}\right)^2\right]-x_i^{(c)}}=\frac{x_i x_n^{(c)}\left(\frac{\alpha_n}{\alpha_j}\right)^2}{x_j\left[1+\left(\frac{\alpha_n}{\alpha_j}\right)^2\right]-x_j^{(c)}} \tag{4.30}$$

da cui, con semplici passaggi,

$$x_j=\frac{x_j^{(c)}}{1-\left(\frac{\alpha_i}{\alpha_j}\right)^2\left(1-\frac{x_i^{(c)}}{x_j}\right)} \quad \forall j : 1 \leqslant j \leqslant n-1 \quad . \tag{4.31}$$

Sostituendo la (4.31) nella (4.29) si ottiene

$$\sqrt{\alpha_n^2\left[\sum_{j=1}^{n-1}\frac{1}{\alpha_j^2}\left(\frac{x_j^{(c)}}{1-\frac{\alpha_i^2}{\alpha_j^2}\left(1-\frac{x_i^{(c)}}{x_j}\right)}\right)^2-\delta+E\right]}=\pm\frac{x_i x_n^{(c)}\left(\frac{\alpha_n}{\alpha_i}\right)^2}{x_i\left[1+\left(\frac{\alpha_n}{\alpha_i}\right)^2\right]-x_i^{(c)}}. \tag{4.32}$$

Quest'ultima espressione è un'equazione a una sola incognita (x_i) facilmente risolvibile e fornisce l'i-esima coordinata di P.

Il carico computazionale può essere ulteriormante semplificato. A questo proposito, si consideri ancora la (4.29); se si indica con G il quadrato del termine a sinistra dell'uguale, si ha

$$G = \frac{x_n^{(c)} \left(\frac{\alpha_n}{\alpha_i}\right)^2}{1 + \left(\frac{\alpha_n}{\alpha_i}\right)^2 - \frac{x_i^{(c)}}{x_i}} = \frac{x_n^{(c)} \left(\frac{\alpha_n}{\alpha_i}\right)^2}{\left(\frac{\alpha_n}{\alpha_i}\right)^2 + y_i} \tag{4.33}$$

dove si è posto $y_i = 1 - \dfrac{x_i^{(c)}}{x_i}$. La (4.33) vale per ogni $i = 1, \ldots, n-1$ dunque, in particolare, per $i = l$. Per questo valore di i la (4.32) diventa

$$\sum_{j=1}^{n-1} \frac{1}{\alpha_j^2} \left(\frac{x_j^{(c)}}{1 - \frac{\alpha_l^2}{\alpha_j^2} y_i} \right)^2 - \delta + E = \frac{{x_n^{(c)}}^2 \alpha_n^2}{\alpha_l^4 \left[\frac{\alpha_n^2}{\alpha_l^2} + y_l\right]^2} \tag{4.34}$$

e fornisce, in generale, $2n$ possibili valori di y_l; ad ognuno di essi corrisponde un valore di G. Esplicitando y_i dalla (4.33) si giunge alla semplice espressione

$$y_i = \left(\frac{\alpha_n}{\alpha_i}\right)^2 \left(\frac{x_n^{(c)}}{G} - 1\right) \tag{4.35}$$

dalla quale noti G e y_l si possono trovare le altre $n-2$ coordinate y_i ($i = 1, \ldots, n-1, i \neq l$). L'ultimo passo è quello di ricavare le coordinate x_i:

$$x_i = \frac{x_i^{(c)}}{1 - y_i} \quad \forall i = 1, \ldots, n-1 \quad \text{e} \quad x_n = \sqrt{\alpha_n^2 \left(\sum_{i=1}^{n-1} \frac{x_i^2}{\alpha_i^2} - \delta + E \right)}\,. \tag{4.36}$$

Questo metodo fornisce $2n$ punti ma l'annullamento delle derivate di $d^2(C', P)$ non è condizione sufficiente per dire che in A si ha un minimo della distanza, potendo anche esservi un massimo o piú in generale un punto di sella.
Comunque, una volta noti i $2n$ punti, essi possono essere ordinati per valori crescenti della distanza effettiva $d(C', A')$; in questo modo il primo di essi corrisponde al un minimo. Nell'appendice A è riportato il programma `radici.m` (scritto in ambiente *Matlab*®); dati $\mathbf{A}$, $\mathbf{b}$ e c e scelto il valore E dell'energia, esso fornisce le soluzioni reali del sistema (4.28).

4.4 Rilassamento geometrico (RG)

Come già accennato in precedenza, in questo algoritmo si comincia col far evolvere la rete da $C = (\frac{1}{2}, \ldots, \frac{1}{2})$. Quando essa raggiunge un punto di minimo $\bar{\mathbf{x}}$, si calcola il valore $\bar{E} = E(\bar{\mathbf{x}})$ e poi si utilizzano le stesse equazioni del paragrafo precedente per trovare punti diversi da $\bar{\mathbf{x}}$ ma con lo stesso valore di energia $\bar{E}$. In questo caso, il fatto di considerare il punto piú vicino al centro aumenta la probabilità che il punto stesso sia interno a Ω'.
In sintesi:

- si fa partire l'evoluzione della rete dal centro $C = (\frac{1}{2}, \ldots, \frac{1}{2})$ dell'ipercubo Ω
- se l'evoluzione porta la rete in un punto di equilibrio $\bar{\mathbf{x}}$ corrispondente a un minimo locale, si calcola l'energia $\bar{E}$ associata a $\bar{\mathbf{x}}$
- si fa il cambio di variabili (4.11) per poter applicare le formule del paragrafo precedente
- trovato il punto A' si opera il cambio di variabili inverso trovando A
- se $A \in \Omega$ si inizia una nuova evoluzione da A; se $A \notin \Omega$ si congiungono A e C con una rette e si fa iniziare la nuova evoluzione dal punto B di intersezione tra questa retta e la frontiera di Ω
- si ripete il procedimento dal secondo punto.

4.5 Complessità dei due algoritmi

Per essere accettata dal punto di vista epistemologico, nell'ambito delle scienze fisiche una nuova teoria deve risultare piú semplice della teoria corrente; analogamente, quando si tratta di risolvere un problema, una metodologia nuova ne soppianta una già esistente se raggiunge risultati della stessa qualità di questa ma in un tempo piú breve.
Qual è il carico computazionale degli algoritmi proposti in questa tesi? Sia nell'IG che nel RG viene richiamato il programma `radici.m` ma per il secondo algoritmo l'operazione è ripetuta un numero di volte che dipende dal

numero di minimi locali che si incontrano prima di raggiungere il minimo globale.
Il numero di questi minimi è indeterminato a priori e perciò risulta difficile classificare il rilassamento geometrico in base alla sua complessità. Per l'IG, al contrario, la ricerca dei punti a distanza minima è effettuata una volta soltanto e quindi, se si considera l'evoluzione della rete come una semplice operazione - ipotesi verosimile nel momento in cui la rete venga realizzata su un supporto fisico - , la complessità si riduce al numero di operazioni elementari compiute dal programma `radici.m`. Leggendone il listato, è facile osservare che il peso computazionale piú rilevante è quello relativo alla ricerca degli autovalori di T e delle radici del polinomio finale di grado $2n$. È stato dimostrato che la complessità di queste due operazioni può essere maggiorata da un polinomio di terzo grado, cioè da n^3 per la prima e da $8n^3$ per la seconda. In totale, quindi, il carico del programma è maggiorato da αn^3. Senza fare nessuna simulazione, quindi, si può già dire che l'IG, si trova in svantaggio nel caso di n-uple supercrescenti per le quali le tecniche tradizionali giungono alla soluzione in un numero di passi pari a n. Nonostante ciò, lo studio proposto in questa sede comincia con le n-uple facili supponendo che, se qualche algoritmo fornisce, per esse, qualche miglioramento rispetto alla semplice evoluzione della rete, ci si possa attendere qualche buon risultato anche per reti relative alla versione difficile del problema del fusto.

4.6 Simulazioni delle reti di Hopfield

4.6.1 Il modello originario

Veniamo ora alle simulazioni effettuate per verificare l'efficacia degli algoritmi proposti. Dei due paradigmi proposti da Hopfield, è stato scelto quello continuo in quanto piú rappresentativo dei sistemi reali; esso è descritto completamente dalle (2.13) e (2.14) che si riportano per comodità:

$$V_i = g_i(u_i) \qquad \text{o anche} \qquad u_i = g_i^{-1}(V_i) \tag{2.13}$$

$$C_i \frac{du_i}{dt} = \sum_{j=1}^{n} T_{ij} V_j - \frac{u_i}{R_i} + I_i \tag{2.14}$$

e pertanto non ci si è preoccupati di simulare un sistema fisico regolato dalle stesse equazioni, come ad esempio un circuito elettrico, ma lo studio è stato limitato alle sole equazioni. Di conseguenza, qualunque programma in grado di simulare equazioni differenziali poteva diventare l'ambiente di lavoro; per le sue caratteristiche di semplicità e di versatilità è stato scelto il pacchetto *Simulink*® utilizzabile in ambiente *Matlab*®. Esso, in particolare, permette di simulare qualunque schema a blocchi lineare e dunque anche l'equazione (2.14). Inoltre, richiamando delle funzioni predefinite in *Matlab*®, si possono inserire nel diagramma anche blocchi non lineari come la funzione di trasferimento (1.3).
Nel capitolo 2 è stata dimostrata la decrescenza di E indipendentemente dai valori di R_i e C_i e dunque, se non interessano i tempi reali di risposta del sistema, si può porre $R_i = C_i = 1 \ \ \forall i = 1, \ldots, n$. Lo schema a blocchi cosí semplificato è quello di Figura 4.8. Nell'eventualità di una realizzazione

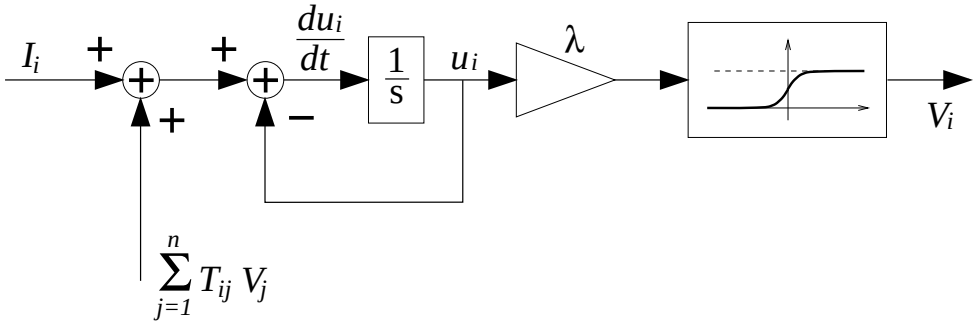


Figura 4.8: Lo schema a blocchi corrispondente alle (2.14) e (2.13).

reale, ad esempio su un supporto fisico di tipo circuitale, i veri valori di C_i e R_i dovranno essere calcolati in base alla velocità di risposta che si vuole ottenere dal sistema ma in ogni caso la costante di tempo $\tau_i = R_i C_i$ dovrà essere compatibile con gli altri parametri del circuito. In particolare, essa dovrà risultare molto maggiore del tempo di salita τ_s del blocco che realizza la funzione non lineare altrimenti l'equazione (2.14), che descrive il comportamento del sistema nel caso $\tau_s = 0$, non è piú valida.
La (2.14), però, descrive il comportamento di un solo neurone e pertanto si

renderebbe necessaria la creazione di n diagrammi a blocchi come quello di Figura 4.8, connessi con n^2 guadagni di valore T_{ij} e ciascuno con una polarizzazione I_i. In realtà *Simulink*® permette simulazioni vettoriali, cioè consente di simulare l'equazione differenziale vettoriale

$$\frac{d\mathbf{u}}{dt} = \mathbf{TV} - \mathbf{u} + \mathbf{I} \quad . \tag{4.37}$$

Lo schema a blocchi che ne deriva è rappresentato in Figura 4.9, dove le linee in grassetto stanno ad indicare il parallelo di n linee. In entrambi gli schemi

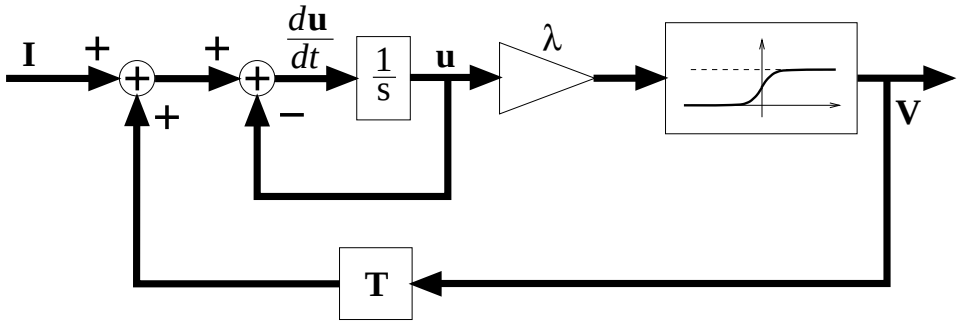


Figura 4.9: Schema a blocchi dell'equazione differenziale vettoriale (4.37).

è presente un blocco di guadagno, indicato con λ, che serve a modificare la pendenza della funzione non lineare.

4.6.2 Il modello semplificato

Nel paragrafo 2.3 è descritto un paradigma neurale che a detta dei suoi inventori presenta le stesse proprietà di memoria associativa del modello di Hopfield. Per verificare questa affermazione, ma anche e soprattutto per confrontare le prestazioni dei due modelli, è stato costruito un diagramma a blocchi anche per il sistema retto dalla (2.23), che riscriviamo per comodità:

$$\frac{d\mathbf{x}}{dt} = \mathbf{Tx} + \mathbf{I} \quad \text{con} \quad -1 \leqslant x_i \leqslant 1 \quad . \tag{2.23}$$

Questo sistema è rappresentato in Figura 4.10, dove per restringere il cam-

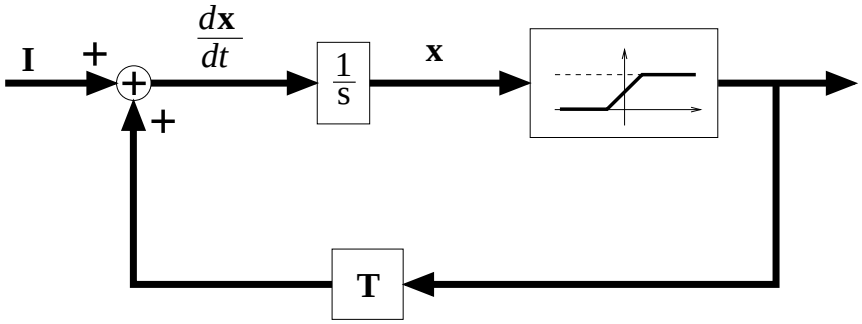


Figura 4.10: Schema a blocchi dell'equazione differenziale vettoriale di Li, Michel e Porod.

po di variabilità di x_i è stato utilizzato un saturatore lineare come nelle simulazioni originali degli autori:

$$F_i(\rho) = \begin{cases} 1 & \text{se } \rho > 1 \\ \rho & \text{se } 0 \leqslant \rho \leqslant 1 \\ 0 & \text{se } \rho < 0 \end{cases} \quad , \quad \forall i : 1 \leqslant i \leqslant n \quad . \tag{4.38}$$

A rigore, la rete dovrebbe essere simulata con $x \in [-1, 1]$; in questo caso, però, cambierebbe la funzione costo (2.72) in quanto ogni neurone con uscita -1 rappresenterebbe un oggetto da sottrarre all'altezza totale o, altrimenti detto, un oggetto con un'altezza negativa, e ciò comporterebbe un cambimento radicale delle modalità di scelta degli oggetti da mettere nel fusto; tra l'altro, la descrizione stessa del problema sarebbe diversa, ammettendo oggetti con altezza negativa. In verità, per avere un comportamento uguale alle reti di Hopfield, è bastato porre uguale a zero il valore inferiore del saturatore di Figura 4.10.

4.6.3 Proprietà di memoria associativa

Costruiti i due schemi a blocchi di Figura 4.8 e Figura 4.10, si è passati a verificare se essi si comportassero effettivamente come memorie associative: per ogni scelta n del numero di neuroni, sono stati generati K vettori di lunghezza n e poi, con la regola di Hopfield (2.2), è stata calcolata la matrice

T, mentre i valori I_i delle polarizzazioni sono stati posti tutti uguali a zero. La scelta del valore di K è stata fatta prendendo spunto in parte da quanto suggerito dallo stesso Hopfield [12] e in parte dalla tabella ricavata nel primo capitolo; in realtà non esiste una regola deterministica che permetta di trovare, per ogni valore di n, un corrispondente valore di K che assicuri l'assoluta assenza di errori e pertanto K è stato scelto con criteri di massima.
Infine, ciascuno dei K vettori da memorizzare è stato assegnato alla rete come stato iniziale di un'evoluzione; quest'ultima, in realtà, doveva coincidere con un'immobilità assoluta dello stato corrente $\mathbf{V}$ della rete, in base all'ipotesi che lo stato iniziale fosse già uno stato di equilibrio. Il verificarsi di quest'ultimo evento era considerato come un successo; il caso contrario, cioè lo spostamento della rete dallo stato in cui veniva posizionata inizialmente, corrispondeva a un fallimento.
Tutte queste operazioni sono realizzate dal programma `verifica.m`, riportato in appendice, che per ciascuna delle K evoluzioni mostra lo stato iniziale e lo stato finale.

4.6.4 Applicazione dei due modelli al problema del fusto

Poiché nella prima fase di verifica le prestazioni di entrambe le reti sono risultate qualitativamente soddisfacenti, si è passati alla seconda fase, ovvero allo studio del comportamento dei due sistemi nell'impiego che interessava questa tesi; è stato scritto un nuovo programma, chiamato `stabili.m` e anch'esso riportato in appendice, che può essere riassunto nei seguenti punti:

- generazione casuale di un'n-upla $\mathbf{a} = (a_1, \ldots, a_n) : 1 \leqslant a_i \leqslant 10n$ e di una soluzione $\mathbf{s} \in \{0, 1\}^n$

- calcolo dell'altezza del fusto $N = \mathbf{a}^\top \mathbf{s}$ e degli elementi di T e di I tramite la (2.74)

- simulazione di 2^n evoluzioni ciascuna caratterizzata da un diverso stato iniziale $\mathbf{V}_0 \in \{0, 1\}^n$

- nel caso in cui lo stato finale $\mathbf{V}_f$ coincida con $\mathbf{V}_0$, cioè fosse uno stato di equilibrio, memorizzazione di $\mathbf{V}_f$ in una matrice $\mathbf{S}$
- visualizzazione della matrice $\mathbf{S}$.

Ciò che si riscontra è un prevedibile aumento del numero degli stati di equilibrio K_e all'aumentare di n. Per esigenze di tempo, però, la simulazione è stata limitata a valori di n non superiori a 20 e non è stato quindi possibile trovare una relazione deterministica, seppur approssimata, tra K_e e le dimensioni della rete.
Ad ogni modo, la soluzione $\mathbf{s}$ compariva tra gli stati di equilibrio nella totalità delle prove effettuate, confermando l'adeguatezza dei sistemi alle esigenze di questo lavoro. I valori di a_i sono stati scelti tra 1 e $10n$ per avere un problema abbastanza difficile, cosí da non incorrere in casi troppo semplici che avrebbero invalidato l'analisi dei risultati. Dopo l'esito positivo di queste verifiche, i due modelli neurali sono stati giudicati validi per l'implementazione vera e propria dei due algoritmi.

Capitolo 5

Discussione dei risultati ottenuti

5.1 Applicazione al fusto supercrescente

La prima fase dello studio, come si è anticipato, riguarda le n-uple supercrescenti, perché esse presentano una certa regolarità di comportamento. Infatti considerando un'n-upla qualunque, i cui valori siano generati a caso, può succedere che, per sbaglio, l'n-upla sia facile quando invece ci si aspetta un caso difficile; per ottenere un'n-upla davvero difficile bisognerebbe procedere come nel cifrario del fusto, ma in tal caso diventerebbe critica la scelta dei valori di u, v e w, e ciò richiederebbe uno studio a parte. Per questo motivo si è deciso di cominciare con un'n-upla supercrescente e precisamente con le potenze di due ponendo $a_i = 2^i$; tra l'altro, un'n-upla siffatta assicura, a differenza di un'n-upla qualunque, l'unicità della soluzione ottima.

5.1.1 Inizializzazione geometrica semplice

Il primo algoritmo studiato è l'IG nella sua versione semplice (IGS), ovvero quella in cui si sceglie, degli m punti ($m \leqslant 2n$) forniti dal metodo della minimazione della distanza, quello piú vicino al centro dell'ipercubo.
I risultati delle evoluzioni, forniti dal programma `fustoiniz.m` che si può trovare in appendice, sono riportati in Tabella 5.1; essi vengono messi a confronto con i risultati registrati nel caso in cui sia assegnato il centro del-

n	IGS	C_s		n	IGS	C_s
5	31/32	30/32		13	677/1000	669/1000
6	60/64	58/64		14	640/1000	641/1000
7	112/128	112/128		15	616/1000	605/1000
8	218/256	212/256		16	559/1000	555/1000
9	414/512	408/512		17	569/1000	560/1000
10	790/1024	778/1024		18	540/1000	529/1000
11	739/1000	725/1000		19	508/1000	502/1000
12	691/1000	680/1000		20	501/1000	489/1000

Tabella 5.1: Confronto tra IG semplice (colonna IGS) ed evoluzione dal centro dell'ipercubo (colonna C_s).

l'ipercubo $C = (\frac{1}{2}, \ldots, \frac{1}{2})$ come stato iniziale dell'evoluzione. Per $n \leqslant 10$ sono state provate tutte le 2^n scelte dell'altezza N del fusto che ammettono soluzione, mentre per n compreso tra 11 e 20 sono stati provati 1000 valori di N scelti a caso tra i 2^n ammissibili.
Gli stessi risultati della Tabella 5.1 sono riportati, per un'analisi piú immediata, nel grafico di Figura 5.1. Da quest'ultima si nota che per quasi tutti i valori di n l'algoritmo migliora le prestazioni della rete ma il miglioramento è quasi trascurabile e non giustifica il carico computazionale associato al programma `radici.m`. Una caratteristica comune di entrambe le curve è una prevedibile diminuzione della percentuale di successi all'aumentare di n.

5.1.2 Inizializzazione geometrica esauriente

Nelle prove discusse nella sezione precedente è stato preso in considerazione solamente il punto piú vicino a C' anche se il metodo dell'annullamento delle derivate fornisce un numero di punti pari a m.
Se si considera ciascuna evoluzione equivalente a un'operazione elementare, il carico computazionale relativo a m tentativi di inizializzazione è ancora polinomiale (si veda il paragrafo 4.5) e quindi appare lecito provare, in sequenza, tutti i punti forniti dalla risoluzione della (4.28) per una stessa istanza del problema; la procedura utilizzata per effettuare questo tipo di prova, defi-

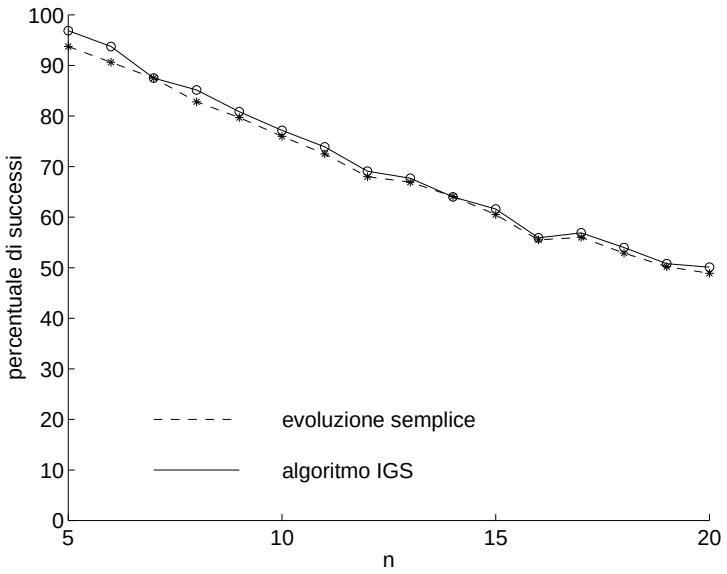


Figura 5.1: Aspetto grafico dei risultati della Tabella 5.1.

nita *inizializzazione geometrica esauriente (IGE)*, e per verificarne l'effcacia consta dei seguenti passi:

- creazione di una lista dei punti trovati, ordinata per distanza da C crescente
- assegnazione del primo punto della lista come stato iniziale della rete per una prima evoluzione, come nel paragrafo precedente; in caso di insuccesso, assegnazione del secondo punto come stato iniziale per una nuova evoluzione, e cosí via
- registrazione, nel caso di successo all'l-esimo tentativo ($l \leqslant m$), del valore di l o, nel caso in cui nessuno degli m punti porti alla soluzione ottima, registrazione dell'insuccesso
- scelta casuale di l punti in Ω da assegnare come stati iniziali per l evoluzioni della rete; in caso di fallimento al passo precedente, $l = m$
- registrazione del successo o del fallimento delle l prove casuali.

In realtà, osservando che nella maggior parte dei casi il centro C dell'ipercubo è un punto di partenza privilegiato rispetto a qualunque altro punto di Ω, nel senso che in media è quello che permette di raggiungere la soluzione ottima nel maggior numero di casi, esso è sempre stato scelto come primo punto della lista casuale. In Tabella 5.2 sono stati riportati i risultati delle operazioni descritte, mentre in Figura 5.2 essi sono in forma grafica.

n	IGE	C_e		n	IGE	C_e
5	32/32	31/32		13	981/1000	852/1000
6	64/64	61/64		14	966/1000	829/1000
7	128/128	122/128		15	966/1000	815/1000
8	256/256	237/256		16	950/1000	790/1000
9	512/512	454/512		17	935/1000	763/1000
10	1019/1024	911/1024		18	919/1000	746/1000
11	991/1000	880/1000		19	891/1000	724/1000
12	978/1000	851/1000		20	887/1000	699/1000

Tabella 5.2: Confronto tra l'IG esauriente (colonna IGE) e le l evoluzioni casuali (colonna C_e).

Si nota un netto miglioramento delle prestazioni che può essere spiegato nel seguente modo: l'algoritmo dell'inizializzazione geometrica si basa sulla supposizione che il punto B, intersezione tra la retta che congiunge A con C e la superficie dell'ipercubo, appartenga al dominio di attrazione del minimo assoluto. Ora, i punti A sono l'immagine, tramite l'inversa della (4.11), dei punti A' e questi, come si diceva, sono in numero di m; pertanto il criterio di scelta che sta alla base dell'IGS, ovvero la scelta del punto A'_1 piú vicino a C', potrebbe non essere ottimale; può darsi, cioè, che si verifichi la situazione di Figura 5.3 dove il punto A'_1 è piú vicino al centro C' rispetto ad A'_2 ma facendo ripartire la rete da B_1 essa finisce in un minimo locale a energia $E_1 > 0$. Per raggiungere il minimo assoluto, invece, si dovrebbe far partire l'evoluzione dal punto B_2. In verità, questa sub-ottimalità è imputabile alla difficoltà intrinseca del problema ed è ragionevole supporre che ogni altro criterio di scelta presenti limitazioni analoghe, senza dimenticare che qualunque

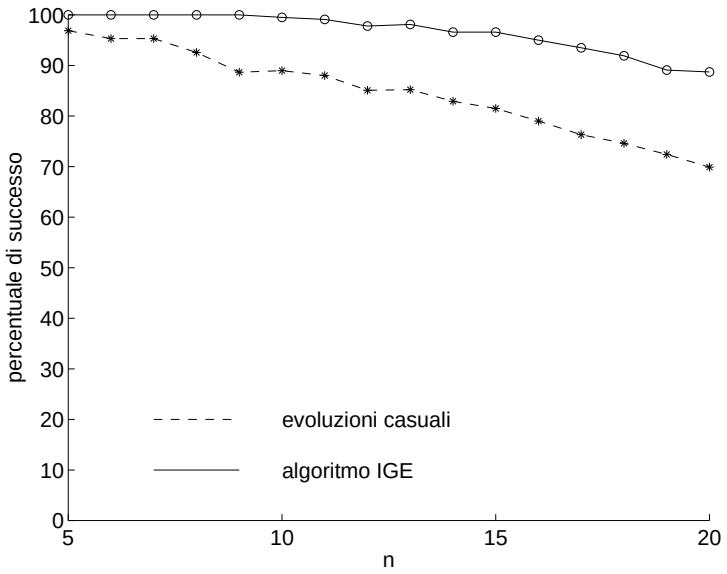


Figura 5.2: Aspetto grafico dei risultati della Tabella 5.2.

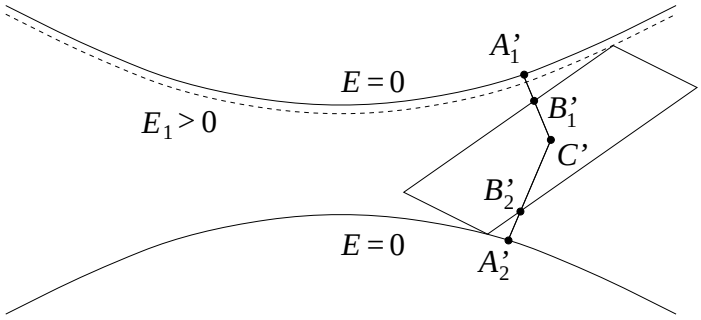


Figura 5.3: Situazione in cui l'IGS fallisce.

strategia comporta, a sua volta, un carico computazionale aggiuntivo.

5.1.3 Efficacia relativa dell'IGE

Volendo fare un'analisi piú dettagliata dei risultati ottenuti, si può costruire, come è stato fatto in Figura 5.4, il grafico di una grandezza che tiene conto di quanto aumentano in percentuale le prestazioni della RN di Hopfield con l'introduzione dell'algoritmo IGE. Tale grandezza, detta *efficacia relativa*, è definita da

$$\eta = 100\frac{IGE - C_e}{C_e} \tag{5.1}$$

dove IGE e C_e indicano le quantità della Tabella 5.2. Come si può notare,

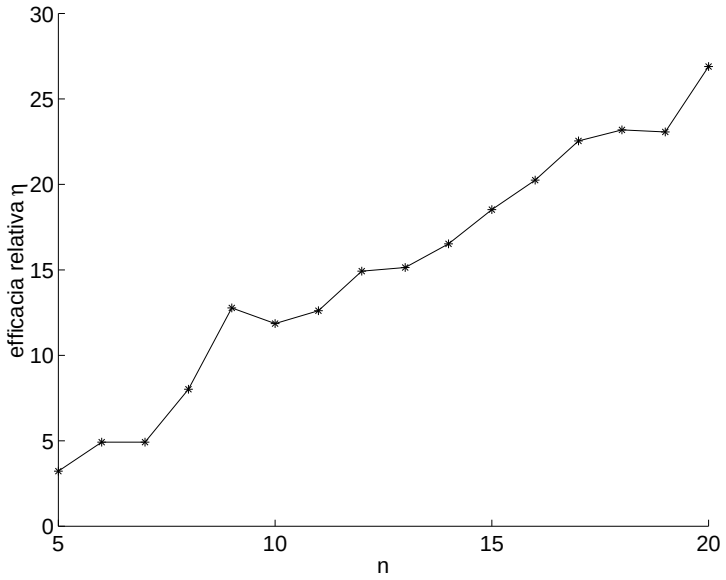


Figura 5.4: Efficacia relativa dell'IGE.

l'efficacia del metodo geometrico aumenta con le dimensioni del problema e per $n = 20$ è maggiore del 25%.

5.1.4 Rilassamento geometrico

L'algoritmo RG si basa sulla supposizione che il metodo dell'annullamento delle derivate, per un iperboloide ad energia $\bar{E}$ maggiore di zero, permetta

di trovare dei punti A interni all'ipercubo Ω aventi la stessa energia $\bar{E}$ del punto di minimo in cui si blocca la rete.
In realtà effettuando diverse simulazioni si è osservato che questo evento non si presenta quasi mai; in altre parole, il valore $\bar{E}$ dell'energia di un minimo locale è quasi sempre talmente piccolo che i punti interni a Ω aventi energia $\bar{E}$ sono pochi e ciò riduce il RG a una semplice iterazione dell'IG.
Per verificare questa affermazione, i risultati del RG sono stati messi a confronto, nella Tabella 5.3, con quelli di una procedura mista che al primo passo assegnava C come stato di partenza, e solo nel caso in cui questa prima evoluzione fallisse procedeva con l'applicazione dell'IG. Come si nota, le due

n	RG	M		n	RG	M
5	32/32	32/32		13	978/1000	976/1000
6	64/64	64/64		14	971/1000	971/1000
7	128/128	128/128		15	959/1000	953/1000
8	256/256	256/256		16	941/1000	940/1000
9	512/512	512/512		17	920/1000	920/1000
10	1019/1024	1019/1024		18	916/1000	910/1000
11	991/1000	996/1000		19	895/1000	894/1000
12	988/1000	983/1000		20	889/1000	883/1000

Tabella 5.3: Confronto tra il rilassamento geometrico (colonna RG) e la procedura mista (colonna M).

colonne differiscono di qualche unità, ma su un totale di 1000 prove simili differenze diventano trascurabili. Per questo motivo lo studio del RG non è proseguito oltre, supponendo che per esso si possano ritenere valide le stesse osservazioni fatte a proposito dell'IG.

5.2 Applicazione al fusto difficile

Dopo aver raccolto i dati incoraggianti del paragrafo precedente sul comportamento dell'inizializzazione geometrica nei confronti di un'n-upla facile, si è passati a verificare se l'algoritmo proposto portasse qualche miglioramento

anche in un caso difficile.
Per generare un'n-upla difficile è stata utilizzata la stessa tecnica indicata nella sezione 4.6.4, e per essere sicuri di non incorrere in casi 'sfortunatamente' semplici per ogni valore di n sono state generate 4 n-uple difficili.
Nella Tabella 5.4 è riportata, per valori di n compresi tra 4 e 10, e per ciascuna delle quattro tecniche del paragrafo 5.1, la media, sulle quattro prove, della percentale di successo. Come si può notare, non vi è sostanziale dif-

n	IGS	C_s	IGE	C_e
4	68,75%	62,50%	100,00%	78,12%
5	39,06%	37,50%	78,91%	64,06%
6	28,12%	29,69%	72,66%	57,81%
7	22,46%	21,48%	57,62%	51,76%
8	10,64%	10,94%	34,67%	42,87%
9	6,20%	5,18%	22,71%	29,54%
10	5,98%	6,35%	21,07%	30,74%

Tabella 5.4: Risultati delle prove relative al fusto difficile.

ferenza tra i valori delle colonne IGS e C_e, mentre analizzando i risultati ottenuti dall'applicazione dell'IGE, ci si accorge che per valori di n minori di 8 esso migliora le prestazioni della semplice rete ma aumentando ancora il numero di neuroni l'algoritmo diventa inutile, se non addirittura dannoso. Per avere una visione immediata di quanto detto, è stato disegnato il grafico di Figura 5.5 nel quale si vede che per $n = 4$ l'efficacia η è circa del 30% ma scende rapidamente per valori di n crescenti, finché per $n = 8$ diventa negativa, ovvero l'algoritmo IGE peggiora le prestazioni della rete semplice.

5.2.1 Giustificazione dei limiti dimostrati

Questo comportamento può essere spiegato partendo dalle stesse considerazioni che nel capitolo 4 hanno portato all'ideazione dell'algoritmo: si supponeva che il punto B non fosse molto distante dalla soluzione ottima ma questa congettura è tanto piú lontana dalla realtà quanto piú è grande il valore di n. Infatti può succedere che il punto B' sia distante dal vertice

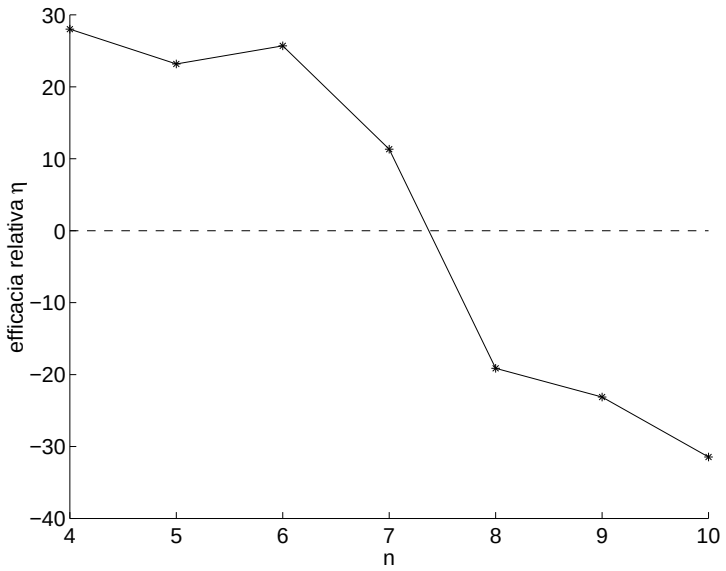


Figura 5.5: Efficacia relativa nel caso difficile.

corrispondente alla soluzione ottima, pur essendo molto vicino al centro del parallelepipedo n-dimensionale, come nell'esempio di Figura 5.6. Per un'a-

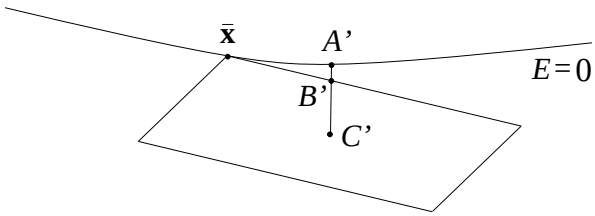


Figura 5.6: Esempio in cui l'algoritmo IG cade in difetto.

nalisi piú precisa, si supponga di essere nel caso peggiore, si supponga cioè che B' coincida con il punto centrale di una faccia $(n-1)$-dimensionale del parallelepipedo, e dette d_v la distanza tra C' e un vertice, e d_f la distanza tra C' e B', si calcoli il rapporto $\frac{d_v}{d_f}$. Senza perdita di generalità, l'operazione può essere ricondotta al caso particolare di un ipercubo di lato unitario

poiché da esso, con trasformazioni lineari, che non modificano i rapporti, si può ottenere qualsiasi parallelepipedo.
La distanza d_v del centro $C = (\frac{1}{2}, \ldots, \frac{1}{2})$ dal vertice $O = (0, \ldots, 0)$ è

$$d_v = \sqrt{\sum_{i=1}^{n} \left(\frac{1}{2} - 0\right)^2} = \frac{\sqrt{n}}{2} \tag{5.2}$$

mentre la distanza dal punto centrale di una faccia, per esempio dal punto $B' = (0, \frac{1}{2}, \ldots, \frac{1}{2})$, è

$$d_f = \sqrt{\left(\frac{1}{2} - 0\right)^2} = \frac{1}{2} \quad ; \tag{5.3}$$

ne segue che

$$\frac{d_v}{d_f} = \sqrt{n} \xrightarrow{n\to\infty} \infty \quad . \tag{5.4}$$

Ciò significa che all'aumentare di n la struttura dell'ipercubo n-dimensionale tende ad assumere una forma 'a riccio' in cui i vertici sono molto piú distanti, dal centro, della maggior parte dei punti di ciascuna faccia. Per questo motivo, quando la forma delle superfici a energia costante si complica a causa della difficoltà del problema, succede che la posizione relativa degli iperboloidi n-dimensionali e dello spazio Ω' è tale per cui anche se un punto A', appartenente all'iperboloide a energia nulla, è vicino a C', il corrispondente punto B' è distante dal vertice associato alla soluzione ottima.
Ciò che si può ancora congetturare, vista la difficoltà di immaginare uno spazio a piú di 3 dimensioni, è che nel caso del fusto facile la regolarità dell'n-upla supercrescente influisca sulla forma dello spazio Ω' e sulla sua posizione rispetto agli iperboloidi facendo sí che l'ipotesi da cui prende spunto l'inizializzazione geometrica sia veritiera. Riguardo all'idea del 'riccio', si può aggiungere, infine, che se da un lato essa aiuta a capire la causa della diminuzione di η nel caso difficile, dall'altro non deve trarre in inganno perché certe proprietà dei parallelepipedi, quale ad esempio la convessità, restano valide per qualunque valore di n.

5.3 Complessità e minimi locali

Come si può notare dai risultati esposti nei paragrafi precedenti, le prestazioni della rete dipendono fortemente dal tipo di problema: se l'n-upla è supercrescente, cioè se il problema del fusto è facile e risolvibile in tempo polinomiale, la rete si comporta bene e anche senza l'applicazione dei metodi geometrici, ovvero scegliendo casualmente il punto di partenza dell'evoluzione, essa trova la soluzione ottima in una percentuale elevata di casi. Al contrario, se l'n-upla è difficile, questa circostanza condiziona la topologia delle superfici n-dimensionali a energia costante, cioè, in pratica, il numero di minimi, diminuendo drasticamente i casi di successo sia della rete semplice sia dell'algoritmo IGE. Si darà ora una giustificazione teorica di questo comportamento, dimostrando che nel caso di un'n-upla supercrescente il numero di stati di equilibrio rispetto all'evoluzione della rete, corrispondenti a punti di minimo di E, è al massimo n. A tal proposito si introdurrà la seguente notazione:

Notazione 5.1. *Date l'n-upla $\mathbf{a} = (a_1, \ldots, a_n)$ e l'altezza N del fusto, si indicherà con $(a_1, \ldots, a_n; N)$, o equivalentemente, con $(\mathbf{a}; N)$, l'istanza del problema e la RN ad essa associata.*

Si consideri, ora, l'equazione (2.30); da essa appare chiaro che se per la coordinata i-esima $\bar{x}_i$ di uno stato $\bar{\mathbf{x}}$ vale

$$\bar{x}_i = 0 \qquad \text{e} \qquad \left.\frac{dx_i}{dt}\right|_{\mathbf{x}=\bar{\mathbf{x}}} \leqslant 0 \quad , \tag{5.5}$$

allora il valore di $\bar{x}_i$ rimane costante; esso infatti tenderebbe a decrescere, essendo la derivata negativa, ma il saturatore ne limita il valore a zero. Analogamente, si può affermare che il valore di $\bar{x}_j$ non cambia nel tempo se

$$\bar{x}_j = 1 \qquad \text{e} \qquad \left.\frac{dx_j}{dt}\right|_{\mathbf{x}=\bar{\mathbf{x}}} \geqslant 0 \quad . \tag{5.6}$$

Sulla base di queste considerazioni si può definire il concetto di punto di equilibrio.

Definizione 5.1. *Un punto $\bar{\mathbf{x}} = (\bar{x}_1, \ldots, \bar{x}_n)^\top \in \Omega$ è un punto di* equilibrio

per la RN se vale

$$\left.\frac{dx_i}{dt}\right|_{\mathbf{x}=\bar{\mathbf{x}}} \begin{cases} \leqslant 0 & \forall i : \bar{x}_i = 0 \\ \geqslant 0 & \forall i : \bar{x}_i = 1 \quad . \end{cases} \tag{5.7}$$

Lemma 5.1. *Sia* $\bar{\mathbf{x}} = (\bar{x}_1, \ldots, \bar{x}_n)^\top$ *un punto di equilibrio per* $(a_1, \ldots, a_n; N)$*; allora* $\bar{\mathbf{x}}_{rid} = (\bar{x}_1, \ldots, \bar{x}_{n-1})^\top$ *è un punto di equilibrio per* $(\mathbf{a}_{rid}; N - \bar{x}_n a_n)$*, dove* $\mathbf{a}_{rid} = (a_1, \ldots, a_{n-1})$*; cioè*

- *se* $\bar{x}_n = 0$, $\bar{\mathbf{x}}_{rid}$ *è un punto di equilibrio per* $(\mathbf{a}_{rid}; N)$
- *se* $\bar{x}_n = 1$, $\bar{\mathbf{x}}_{rid}$ *è un punto di equilibrio per* $(\mathbf{a}_{rid}; N - a_n)$.

Dimostrazione. Si supponga $\bar{x}_n = 0$; si deve dimostrare che per la rete $(\mathbf{a}_{rid}; N)$ vale

$$\left.\frac{dx_i}{dt}\right|_{\mathbf{x}=\bar{\mathbf{x}}_{rid}} \begin{cases} \leqslant 0 & \forall i : \bar{x}_i = 0 \\ \geqslant 0 & \forall i : \bar{x}_i = 1 \quad . \end{cases} \tag{5.8}$$

Definite $\mathbf{T}' = -\mathbf{a}_{rid}^\top \mathbf{a}_{rid}$ e $I_i' = \dfrac{a_i^2}{2} + N a_i \quad \forall i = 1, \ldots, n-1$, dalla (2.23) si ha $\forall i = 1, \ldots, n-1$,

$$\left.\frac{dx_i}{dt}\right|_{\mathbf{x}=\bar{\mathbf{x}}_{rid}} = \underline{T'}_i \bar{\mathbf{x}} + I_i' = T_{i1}' \bar{x}_1 + \cdots + T_{i,n-1}' \bar{x}_{n-1} + I_i' \quad . \tag{5.9}$$

Ma $T_{ij}' = -a_i a_j = T_{ij}$ e $I_i' = I_i$; dunque, essendo $\bar{x}_n = 0$,

$$\left.\frac{dx_i}{dt}\right|_{\mathbf{x}=\bar{\mathbf{x}}_{rid}} = T_{i1}\bar{x}_1 + \cdots + T_{i,n-1}\bar{x}_{n-1} + T_{in}\bar{x}_n + I_i = \left.\frac{dx_i}{dt}\right|_{\mathbf{x}=\bar{\mathbf{x}}} \tag{5.10}$$

e la (5.8) è vera per l'ipotesi che $\bar{\mathbf{x}}$ fosse di equilibrio per $(\mathbf{a}; N)$.
Sia ora $\bar{x}_n = 1$; si tratterà di dimostrare che la (5.8) vale per $(\mathbf{a}_{rid}; N - a_n)$. In questo caso si definiscono $\mathbf{T}'' = -\mathbf{a}_{rid}^\top \mathbf{a}_{rid} = \mathbf{T}'$ e $I_i'' = \dfrac{a_i^2}{2} + (N - a_n)a_i = I_i' - a_n a_i$. Pertanto, $\forall i = 1, \ldots, n-1$,

$$\left.\frac{dx_i}{dt}\right|_{\mathbf{x}=\bar{\mathbf{x}}_{rid}} = \underline{T''}_i \bar{\mathbf{x}} + I_i'' = T_{i1}'' \bar{x}_1 + \cdots + T_{i,n-1}'' \bar{x}_{n-1} - a_n a_i + I_i' \quad . \tag{5.11}$$

Ma $\bar{x}_n = 1$ e perciò $-a_n a_i = T_{in}\bar{x}_n$; quindi

$$\left.\frac{dx_i}{dt}\right|_{\mathbf{x}=\bar{\mathbf{x}}_{rid}} = T_{i1}\bar{x}_1 + \cdots + T_{i,n-1}\bar{x}_{n-1} + T_{in}\bar{x}_n + I_i = \left.\frac{dx_i}{dt}\right|_{\mathbf{x}=\bar{\mathbf{x}}} \tag{5.12}$$

e ancora una volta la (5.8) è vera per ipotesi. □

Questo risultato comporta che gli stati di equilibrio di una rete di n neuroni, strutturata per la risoluzione di un'istanza del problema del fusto, possano essere ottenuti dagli stati di equilibrio di due reti di $n-1$ neuroni; la prima di esse corrisponde all'istanza ridotta $(\mathbf{a}_{rid}; N)$ e la seconda all'istanza $(\mathbf{a}_{rid}; N - a_n)$.

Teorema 5.1. *Si supponga che l'evoluzione della rete* $(\mathbf{a}; N)$ *la porti in uno stato di equilibrio* $\bar{\mathbf{x}}$ *tale che* $\mathbf{a}\bar{\mathbf{x}} = N_1 < N$. *Allora*

- $\forall i : x_i = 0 \quad$ *è* $\quad \dfrac{a_i}{2} \geqslant N - N_1$
- $\forall i : x_i = 1 \quad$ *è* $\quad \dfrac{a_i}{2} \leqslant N - N_1$.

Dimostrazione. Supponiamo, per assurdo, che sia $\bar{x}_i = 0$ con $\dfrac{a_i}{2} < N - N_1$. Se cosí fosse, dovrebbe essere

$$\left.\frac{dx_i}{dt}\right|_{\mathbf{x}=\bar{\mathbf{x}}} = \underline{T}_i \bar{x} + I_i \leqslant 0 \tag{5.13}$$

cioè

$$\begin{aligned} -a_i a_1 \bar{x}_1 - a_i a_2 \bar{x}_2 - \ldots - a_i a_{i-1} \bar{x}_{i-1} - a_i a_{i+1} \bar{x}_{i+1} - \ldots \\ \ldots - a_i a_n \bar{x}_n + N a_i - \frac{a_i}{2} \leqslant 0 \quad . \end{aligned} \tag{5.14}$$

Essendo $\bar{x}_i = 0$, la (5.14) equivale alla

$$-a_i(a_1 \bar{x}_1 + \cdots + a_n \bar{x}_n) + N a_i - \frac{a_i^2}{2} \leqslant 0 \quad ; \tag{5.15}$$

ma allora dovrebbe valere

$$N a_i - N_1 a_i - \frac{a_i^2}{2} \leqslant 0 \qquad \text{ovvero} \qquad N - N_1 \leqslant \frac{a_i}{2} \tag{5.16}$$

che contraddice l'ipotesi assurda. Analogamente si dimostra la seconda parte del teorema. □

Corollario 5.1. Se $N \geqslant \sum_{i=1}^{n} a_i$ l'unico stato di equilibrio è $\bar{\mathbf{x}}_{uni} = (1, \ldots, 1)^\top$.

Infatti, se per assurdo fosse $\bar{x}_j = 0$ per qualche j, ciò significherebbe che

$$\mathbf{a}\bar{\mathbf{x}} = N_1 \leqslant \sum_{i=1}^{n} a_i - a_j \tag{5.17}$$

e ciò comporterebbe

$$\frac{a_j}{2} < a_j \leqslant \sum_{i=1}^{n} a_i - N_1 \leqslant N - N_1 \tag{5.18}$$

cioè

$$\frac{a_j}{2} < N - N_1 \tag{5.19}$$

che contraddice la tesi del teorema precedente. Ammettendo che N possa assumere anche valori negativi, in modo analogo si dimostra che se $N < 0$ l'unico punto di equilibrio è $\bar{\mathbf{x}}_{zeri} = (0, \ldots, 0)^\top$.

Teorema 5.2. *Sia data l'n-upla supercrescente* $\mathbf{a} = (a_1, \ldots, a_n)$. *Qualunque sia* N, *il numero degli stati di equilibrio di* $(\mathbf{a}; N)$ *è al piú* n.

Dimostrazione. Si definiscano innanzitutto i seguenti insiemi:

$$S_0 = \left\{\bar{\mathbf{x}} = (\bar{x}_1, \ldots, \bar{x}_n)^\top : \bar{\mathbf{x}}_{rid} \text{ è di equilibrio per } (\mathbf{a}_{rid}; N) \text{ e } \bar{x}_n = 0\right\}$$

$$S_1 = \left\{\bar{\mathbf{x}} = (\bar{x}_1, \ldots, \bar{x}_n)^\top : \bar{\mathbf{x}}_{rid} \text{ è di equilibrio per } (\mathbf{a}_{rid}; N - a_n) \text{ e } \bar{x}_n = 1\right\}.$$

Per il lemma 5.1, l'insieme S degli stati di equilibrio di $(\mathbf{a}; N)$ è un sottoinsieme di $S_0 \cup S_1$. Si supponga, ora, che sia $N \geqslant a_n$; per la supercrescenza di $\mathbf{a}$ è anche $N \geqslant \sum_{i=1}^{n-1} a_i$ e dunque, per il corollario appena visto, $(\mathbf{a}_{rid}; N)$ ha come unico punto di equilibrio $(1, \ldots, 1)^\top$ il che equivale a dire che S_0 ha un solo elemento. Se, d'altro canto, $N \leqslant a_n$, allora è $N - a_n \leqslant 0$ e dunque $(\mathbf{a}_{rid}; N - a_n)$ ha come unico punto si equilibrio $(0, \ldots, 0)^\top$ il che equivale a dire che S_1 ha un solo elemento. In entrambi i casi, indicata con $\mathfrak{C}(n)$ la cardinalità del generico insieme S i cui elementi hanno dimensione n, si ha $\mathfrak{C}(n) \leqslant 1 + \mathfrak{C}(n-1)$. Procedendo in questo modo per valori discendenti di n si giunge alla tesi. □

Il teorema appena dimostrato sfrutta la supercrescenza dell'n-upla $\mathbf{a}$ e dunque non ha validità generale; tuttavia è possibile utilizzare lo stesso procedimento per trovare una quantità che limita superiormente il numero di minimi di un'n-upla $\mathbf{b}$ qualunque. Si supponga, senza perdita di generalità, che $\mathbf{b}$ sia ordinata, cioè che $b_1 \leqslant b_2 \leqslant \ldots \leqslant b_n$; in primo luogo, il Corollario 5.1

permette di affermare che se $N \leqslant 0$ oppure $N \geqslant \sum_{i=1}^{n} b_i$ la rete associata all'istanza $\mathcal{I} = (\mathbf{b}; N)$ ha un solo punto di equilibrio. Affinché i punti di equilibrio siano 2 o piú, deve valere

$$0 < N < \sum_{i=1}^{n} a_i \quad . \tag{5.20}$$

Se la (5.20) è verificata, si può operare con la tecnica di bipartizione suggerita dal Lemma 5.1: si ricavano i punti di equilibrio di $(\mathbf{b}; N)$ dai punti di equilibrio delle reti ridotte associate alle istanze $\mathcal{I}_0 = (\mathbf{b}_{rid}; N) = (b_1, \ldots, b_{n-1}; N)$ e $\mathcal{I}_1 = (\mathbf{b}_{rid}; N - b_n)$. Applicando ancora una volta il Corollario 5.1 a $\mathcal{I}_0$ e $\mathcal{I}_1$, si trova che

- $\mathcal{I}_0$ ha piú di un punto di equilibrio se $0 < N < \sum_{i=1}^{n-1} b_i$,
- $\mathcal{I}_1$ ha piú di un punto di equilibrio se $0 < N - b_n < \sum_{i=1}^{n-1} b_i$, cioè se $b_n < N < \sum_{i=1}^{n} b_i$.

Affinché il problema non ammetta semplificazioni, per entrambe le istanze i punti di equilibrio devono essere piú di due, cioè deve valere

$$b_n < N < \sum_{i=1}^{n-1} b_i \quad . \tag{5.21}$$

Se la (5.21) è verificata, si bipartiscono $\mathcal{I}_0$ e $\mathcal{I}_1$ trovando

- $\mathcal{I}_{00} = (b_1, \ldots, b_{n-2}; N)$ che ammette piú di un punto di equilibrio se $0 < N < \sum_{i=1}^{n-2} b_i$,
- $\mathcal{I}_{01} = (b_1, \ldots, b_{n-2}; N - b_{n-1})$ che ammette piú di un punto di equilibrio se $b_{n-1} < N < \sum_{i=1}^{n-1} b_i$,
- $\mathcal{I}_{10} = (b_1, \ldots, b_{n-2}; N - b_n)$ che ammette piú di un punto di equilibrio se $b_n < N < \sum_{i=1}^{n-2} b_i + b_n$,

- $\mathcal{I}_{11} = (b_1, \ldots, b_{n-2}; N - b_n - b_{n-1})$ che ammette piú di un punto di equilibrio se $b_{n-1} + b_n < N < \sum_{i=1}^{n} b_i$.

Affinché neache in questo caso il problema ammetta semplificazioni, devono essere valide contemporaneamente tutte e quattro le disequazioni scritte, cioè deve essere

$$b_n + b_{n-1} < N < \sum_{i=1}^{n-2} b_i \quad . \tag{5.22}$$

È facile vedere che, procedendo in questa direzione, ad ogni passo la disequazione (5.22) si modifica con il passaggio di un termine b_j dalla sommatoria di destra a quella di sinistra. Al generico passo k-esimo ($k \geqslant 1$) ci si trova ad avere 2^k reti di dimensione $n - k$ e affinché il problema non ammetta semplificazioni deve essere

$$b_n + b_{n-1} + \cdots + b_{n-k+1} < N < \sum_{i=1}^{n-k} b_i \quad . \tag{5.23}$$

La (5.23) non può essere valida per un numero di passi idefinito perché per $k = \bar{k} = n - \lfloor \frac{n}{2} \rfloor$ diventa

$$b_n + b_{n-1} + \cdots + b_{\lfloor \frac{n}{2} \rfloor + 1} < N < \sum_{i=1}^{\lfloor \frac{n}{2} \rfloor} b_i \tag{5.24}$$

ed è chiaro che, essendo $\mathbf{b}$ ordinata, non si può proseguire oltre; quindi delle $2^{\bar{k}+1}$ reti del passo successivo di bipartizione, almeno una ammette un solo punto di equilibrio. Se si suppone, per non appesantire troppo il discorso, che n sia pari, si trova $\bar{k} = \frac{n}{2}$ e con le notazioni del Teorema 5.2 si arriva alla formula ricorsiva

$$\mathfrak{C}(n) \leqslant \left(2^{\frac{n}{2}} - 1\right) \mathfrak{C}\left(\frac{n}{2}\right) + 1 \quad . \tag{5.25}$$

5.4 Conclusioni

Ogni volta che si costruisce una rete neurale per risolvere un problema di natura combinatoria la difficoltà principale che si deve superare per consentire alla rete di fornire la soluzione ottima è l'eliminazione, o eventualmente

l'aggiramento, dei punti di equilibrio corrispondenti a soluzioni non ottime. Il problema del fusto non fa eccezione e lo scopo principale di questa tesi era quello di proporre un nuovo algoritmo per evitare lo scoglio dei minimi locali ricavando un punto privilegiato, appartenente al dominio di attrazione del minimo assoluto, da cui far iniziare l'evoluzione della rete. Successivamente, con diversi tipi di prove, è stata controllata l'efficacia di tale algoritmo. Ciò che si è potuto constatare è un buon miglioramento delle prestazioni della semplice rete nel caso di istanze facili ma un comportamento opposto nei confronti di istanze difficili. Qualitativamente si può affermare che la difficoltà del problema influisce sulle capacità risolutive della rete e nel paragrafo precedente si è giustificata questa affermazione dimostrando che nel caso di un'ennupla supercrescente il numero di punti di equilibrio è minore del numero di neuroni: è lecito, quindi, aspettarsi che per trovare la soluzione ottima siano sufficienti pochi tentativi, anche scegliendo a caso il punto di partenza dell'evoluzione. Al contrario, nel caso di ennuple qualunque, cioè in particolare per ennuple difficili, l'unica maggiorazione trovata è di tipo esponenziale e perciò anche un algoritmo infallibile, che facesse uscire la rete da ogni minimo locale in cui essa incorresse durante la sua evoluzione, andrebbe applicato un elevato numero di volte.
Se ne può concludere che le reti neurali artificiali, utili se usate come memorie associative, nel campo dell'ottimizzazione combinatoria non portano vantaggi considerevoli rispetto agli algoritmi tradizionali, e ciò cosituisce un'ulteriore conferma della congettura $P \neq NP$.

Appendice

Si riporta qui di seguito il programma `radici.m` con cui si ricavano i $2n$ punti dei due metodi del capitolo 4. Esso serve per trovare il polinomio corrispondente alla (4.32). Moltiplicando per $\prod_{j=1}^{n-1} \alpha_j^2 \left(1 - \frac{\alpha_l^2}{\alpha_j^2} y_l\right)^2$, dalla stessa (4.32) si trova

$$\sum_{j=1}^{n-1} {x_j^{(c)}}^2 \left[\prod_{\substack{k=1 \\ k\neq j}}^{n-1} \alpha_k^2 \left(1 - \frac{\alpha l^2}{\alpha_k^2} y_l\right)^2\right] + (E-\delta) \prod_{j=1}^{n-1} \alpha_j^2 \left(1 - \frac{\alpha_l^2}{\alpha_j^2} y_l\right)^2 =$$

$$= \frac{{x_n^{(c)}}^2 \alpha_n^2}{\alpha_l^4 \left[\frac{\alpha_n^2}{\alpha_l^2} + y_l\right]^2} \prod_{j=1}^{n-1} \alpha_j^2 \left(1 - \frac{\alpha_l^2}{\alpha_j^2} y_l\right)^2 \tag{26}$$

da cui, moltiplicando per $\alpha_l^4 \left[\frac{\alpha_n^2}{\alpha_l^2} + y_l\right]^2$,

$$\alpha_l^4 \left[\frac{\alpha_n^2}{\alpha_l^2} + y_l^2\right]^2 \left(\sum_{j=1}^{n-1} {x_j^{(c)}}^2 \left[\prod_{\substack{k=1 \\ k\neq j}}^{n-1} \alpha_k^2 \left(1 - \frac{\alpha_l^2}{\alpha_k^2} y_l\right)^2\right]\right) +$$

$$+\alpha_l^4 \left[\frac{\alpha_n^2}{\alpha_l^2} + y_l\right]^2 (E-\delta) \prod_{j=1}^{n-1} \alpha_j^2 \left(1 - \frac{\alpha_l^2}{\alpha_j^2} y_l\right)^2 =$$

$$= {x_n^{(c)}}^2 \alpha_n^2 \prod_{j=1}^{n-1} \alpha_j^2 \left(1 - \frac{\alpha_l^2}{\alpha_j^2} y_l\right)^2 \tag{27}$$

che è un polinomio di grado $2n$ le cui radici sono i possibili valori di y_i. Trovato il generico punto $A = (V_1^A, \ldots, V_n^A)$, se esso non sta in Ω, si consideri

la retta passante per C e A:

$$\begin{cases} V_1 &= \frac{1}{2} - t(\frac{1}{2} - V_1^A) \\ \vdots & \\ V_n &= \frac{1}{2} - t(\frac{1}{2} - V_n^A) \quad . \end{cases} \tag{28}$$

Il punto B è dato dall'intersezione di questa retta con la faccia dell'ipercubo definita da $V_k = cost$ dove $V_k : |x_k - \frac{1}{2}| = \max_i |x_i - \frac{1}{2}|$ e $cost$ vale 0 o 1 a seconda della posizone del punto: posto, dunque,

$$cost = V_k^{(c)} - t(V_k^{(c)} - V_k^A) \tag{29}$$

si ricava il valore di t:

$$t = \frac{V_k^{(c)} - cost}{V_k^{(c)} - x_k^A} \quad . \tag{30}$$

```
% trova le soluzioni con il metodo della minima distanza
function r = radici(A,b,c,E)
[U,D] = eig(A);
d = U'*b;
p = -diag(D);
n = size(A,1);
p(n,1) = -p(n,1);
alfa = sqrt(2./p);
K = -c + E + sum((d.^2)./(2*diag(D)));
xcentro = U'*(1/2)*ones(n,1) + d./diag(D);
y = zeros(n-1,1);   % coordinate del punto a minima distanza

coord = 1;   % coordinata scelta per calcolare G
Q = 0;

for j = 1:n-1,
   P = 1;
   for k = 1:n-1,
```

```
      if k ~= j,
         pk = [-((alfa(coord,1)/alfa(k,1))^2) 1];
         pk = conv(pk,pk);
         pk = (alfa(k,1)^2) * pk;
         P = conv(P,pk);
      end;
   end;
   P = (xcentro(j,1)^2)*P;
   Q = somma(Q,P);
end;
Q = alfa(n,1)^2 * Q;
pn = [(alfa(coord,1)/alfa(n,1))^2 1];
pn = conv(pn,pn);
Q = conv(pn,Q);

R = 1;
for k = 1:n-1,
   pk = [-((alfa(coord,1)/alfa(k,1))^2) 1];
   pk = conv(pk,pk);
   pk = alfa(k,1)^2 * pk;
   R = conv(R,pk);
end;
P = (xcentro(n,1)^2)*R;
Q = somma(Q,-P);

R = alfa(n,1)^2 * R;
pn = [(alfa(coord,1)/alfa(n,1))^2 1];
pn = conv(pn,pn);
R = conv(pn,R);
R = K * R;
Q = somma(Q,R);

soly = roots(Q);
% bisogna scartare le soluzioni complesse:
```

```
y_reali = find(imag(soly) == 0);
if isempty(y_reali),
   input('Non ci sono soluzioni reali');
else
   soly = soly(y_reali(:,1),1);
end;
% la matrice delle possibili soluzioni
% (ogni colonna e' un punto) e'
X = zeros(n,size(y_reali,1));
X_reali = [];
beta = (alfa(n,1)/alfa(coord,1))^2;
for i = 1:size(y_reali,1),
   y(coord,1) = soly(i,1);
   G = (xcentro(n,1)*beta) / (beta + y(coord,1));
   for j = 1:n-1,
      if j ~= coord,
         y(j,1) = (alfa(n,1)/alfa(j,1))^2*(xcentro(n,1)/G-1);
      end;
   end;
   X(1:n-1,i) = xcentro(1:n-1,1) ./ (1-y);
   X(n,i) = sqrt( alfa(n,1)^2 * (sum( (X(1:n-1,i).^2) ./
                              ./(alfa(1:n-1,1).^2) ) + K) );
   if imag(X(n,i)) == 0,
      X_reali(:,size(X_reali,2)+1) = X(:,i);
   end;
end;

F = zeros(size(X_reali));
for i = 1:size(F,2),
   F(:,i) = U*(X_reali(:,i) - d./diag(D));
   en(1,i) = (1/2)*F(:,i)'*A*F(:,i) + F(:,i)'*b + c;
end;
r = F;
```

La funzione `somma`, richiamata dal programma alle righe 27, 43 e 51, esegue la somma di due vettori di dimensioni diverse:

```
function s = somma(a,b)
q = zeros(1,max(size(a,2),size(b,2)));
q2 = size(q,2);
a2 = size(a,2);
b2 = size(b,2);
q(1,q2-a2+1:q2) = a;
q(1,q2-b2+1:q2) =  q(1,q2-b2+1:q2) + b;

s = q;
```

Una volta ottenuti i punti utili, l'algoritmo procede con la verifica della loro ottimalità. Questa seconda fase è eseguita dal programma `fustoiniz.m`:

```
% applica il metodo dell'inizializzazione geometrica

corrette = zeros(16,4);
sbagliate = zeros(16,4);
for n = 5:20
for casi = 1:4,
if casi > 2,
   tent = 0;
else
   tent = 1;
end;
a = 2.^(0:1:n-1);
if or(casi == 1,casi ==3)
   modal = 1;
else
   modal = 0;
end;
tent = 0;
modal = 1;
```

```
% nel caso difficile si aggiungano
% le seguenti tre righe

%if and(modal,tent)
   %a = ceil(rand(1,n)*10*n);
%end;

centro = (1/2)*ones(n,1);

sol = zeros(1,n)           % soluzione
iter = 1;
if and(modal,tent),
   solsol = [];
end;
if modal
   prove = [];
end;
while sol(1,1) ~= 2,
   k = 1
   if and(modal,tent)
      solsol(iter,1:n) = sol;
   end;
   N = sol*a';
   V_iniziali = [];
   if modal
      Evin = [];
   else
      Evin = ones(1,prove(1,iter));
   end;
   T = -a'*a;
   T = T - diag(diag(T));     % matrice delle connessioni
   I = (-(a'.^2)/2 + N*a');   % vettore delle polarizzazioni
   c = (N^2)/2;
```

```
if modal
   V = radici(-T,-I,c,0);
   metodo = 1;
   lont = [];
   for i = 1:size(V,2),
      punto = V(:,i);
      lont(1,i) = sqrt( sum( (centro-punto).^2 ) );
      ordlont = sort(lont);
      dist = abs(centro - punto);
      % c_p_d = coordinata piu' distante
      c_p_d = min(find(dist == max(dist)));
      % il piano con cui trovare l'intersezione e':
      cost = (1/2)*(1+sign(punto(c_p_d,1)));
      t = ((1/2)-cost) / ((1/2)-punto(c_p_d,1));
      Vin = centro - t*(centro - punto);

      % poi lo modifico per poterlo utilizzare
      Vin(find(Vin==1)) = 1-1e-16;
      Vin(find(Vin==0)) = 1e-16;
      Evin(1,i) = -(1/2)*Vin'*T*Vin - Vin'*I + c;
      V_iniziali(:,i) = Vin;

      end;
   end;
while 1,
   if modal
      scelta = min(find(lont==ordlont(1,k-1)))
      Vin = V_iniziali(:,scelta);
      V_iniziale = Vin;
   else
      if k == 1,
         V_iniziale = (1/2)*ones(1,n);
      else
         V_iniziale = rand(1,n);
```

```
      end;
   end;

   [tempo1,stat1,Y1] = sim('LMP');  % simulazione

   if isequal(Y1(end,:),sol),
      corrette(n-4,casi) = corrette(n-4,casi) + 1;
      prove(1,iter) = k;
      break;
   end;
   k = k + 1;
   if tent
      if k == 2, k = size(Evin,2) + 1; end;
   end;
   if k-1 > size(Evin,2),
      sbagliate(n-10,notte) = sbagliate(n-10,notte) +1;
      prove(1,iter) = size(Evin,2);
      break
   end;

end;

if n <= 10
   sol(1,n) = sol(1,n) + 1;
   for j = n:-1:2,
      if sol(1,j) == 2,
         sol(1,j) = 0;
         sol(1,j-1) = sol(1,j-1) + 1;
      end;
   end;
   sol
   iter = iter + 1;
else
   sol = round(rand(1,n));
```

```
      iter = iter + 1
      if iter == 1001; break; end;
   end;

end;  % while

end;  % casi
end;  % n
corrette
sbagliate
```

Bibliografia

[1] Silvio Cammarata: *"Reti neuronali"*, Etaslibri, Milano, 1990.

[2] Roger Penrose: *"La mente nuova dell'imperatore"*, Rizzoli, Milano, 1992.

[3] Antonio D'Amore, Paolo Inchingolo: *"Problemi di decodifica dell'attività nervosa unitaria interpretata quale modulazione non lineare di posizione"*, 1980.

[4] W. McCulloch, W. Pitts: *"A logical calculus of the ideas immanent in nervous activity"*, *Bull. Math. Biophys.* 5, 1943, pp. 115-133. Ristampato in *"Neurocomputing"* a cura di J. A. Anderson e E. Rosenfeld, MIT Press, Cambridge, 1988.

[5] F. Rosenblatt: *"The perceptron: a probabilistic model for information storage and organization in the brain"*, *Psych. Rev.* 65, 1958, pp. 386-408. Ristampato in *"Neurocomputing"* a cura di J. A. Anderson e E. Rosenfeld, MIT Press, Cambridge, 1988.

[6] B. Widrow, M. Hoff: *"Adaptive switching circuits"*, *1960 IRE WESCON Conv. Rec.*, New York: IRE, pp. 96-104. Ristampato in *"Neurocomputing"* a cura di J. A. Anderson e E. Rosenfeld, MIT Press, Cambridge, 1988.

[7] M. Minsky, S. Papert: *"Perceptrons"*, MIT Press, Cambridge, 1969.

[8] D. A. Sprecher: *"On the structure of continuous functions of several variables"*, *Trans. Am. Math. Soc.* 115, 1965, pp. 340-555.

[9] A. Mazzetti: *"Reti Neurali Artificiali"*, Apogeo, Milano, 1991.

[10] D. E. Rumelhart, G. E. Hinton, R. J. Williams: *"Learning internal representations by error propagation"*, Parallel Distributed Processing: Exploration in the Microstructures of Cognition, Vol. 1, MIT Press, Cambridge, 1986, pp. 318-362. Ristampato in *"Neurocomputing"* a cura di J. A. Anderson e E. Rosenfeld, MIT Press, Cambridge, 1988.

[11] T. Kohonen: *"Correlation matrix memories"*, *IEEE Trans. Comput.*, C-21, 1972, pp. 353-359. Ristampato in *"Neurocomputing"* a cura di J. A. Anderson e E. Rosenfeld, MIT Press, Cambridge, 1988.

[12] J. J. Hopfield: *"Neural networks and physical systems with emergent collective computational abilities"*, (1982) *Proc. Nat. Acad. Sci.* vol. 79, pp. 2554-2558. Ristampato in *"Artificial Neural Networks"* a cura di E. Sánchez-Sinencio e C. Lau, IEEE Press, New York, 1992.

[13] B. Kosko: *"Bidirectional Associative Memories"*, *IEEE Trans. Syst. Man. Cybern.* vol. 18, no. 1, Gen./Feb. 1988, pp. 49-60. Ristampato in *"Artificial Neural Networks"* a cura di E. Sánchez-Sinencio e C. Lau, IEEE Press, New York, 1992.

[14] David J. C. MacKay: *"Information Theory, Pattern Recognition and Neural Networks"* in *http://wol.ra.phy.cam.ac.uk/mackay/itprnn*, 1998.

[15] J. J. Hopfield: *"Neurons with graded response have collective computational properties like those of two-state neurons"*, *Proc. Nat. Acad. Sci.* 81, 1984, pp. 3088-3092. Ristampato in *"Neurocomputing"* a cura di J. A. Anderson e E. Rosenfeld, MIT Press, Cambridge, 1988.

[16] J. H. Li, A. N. Michel, W. Porod: *"Qualitative Analysis and Synthesis of a Class of Neural Networks"*, *IEEE Trans. Circuits Syst.* vol. 35, no. 8, Ago. 1988, pp. 976-986.

[17] J. H. Li, A. N. Michel, W. Porod: *"Analysis and Sybthesis of a Class of Neural Networks: Linear Systems Operating on a Closed Hypercube"*, *IEEE Trans. Circuits Syst.* vol. 36, no. 11, Nov. 1989, pp. 1405-1422.

[18] Michael R. Garey, David S. Johnson: *"COMPUTERS AND INTRACTABILITY. A Guide to the Theory of NP-Completeness"*, *W. H. Freeman*, San Francisco, 1979.

[19] Jehoshua Bruck, Joseph W. Goodman: *"On the Power of Neural Networcks for Solving Hard Problems"*, *IEEE Neural Information Processing Systems Conference*, Denver, Colorado, Novembre 1987. Ristampato in Journal of Complexity, vol. 6, 1990, pp. 129-135.

[20] J. J. Hopfield, D. W. Tank: *""Neural" Computation of Decision in Optimization Problems"*, *Biol. Cybern.* 52, 1985, pp. 141-152.

[21] D. W. Tank, J. J. Hopfield: *"Simple "Neural" Optimization Networks: An A/D Converter, Signal Decision Circuit, and a Linear Programming Circuit"*, *IEEE Trans. Circuits Syst.* vol. CAS-33, no. 5, Mag. 1986, pp. 533-541.

[22] Jehoshua Bruck, Mario Blaum: *"Neural Networks, Error-Correcting Codes, and Polynomials over the Binary* n-*Cube"*, *IEEE Trans. Inform. Theory* vol. 35, no. 5, Set. 1989, pp. 976-987.

[23] Giuseppe Longo: *"Teoria dell'informazione"*, Boringhieri, Torino, 1980.

[24] Francesco Fabris, Giacomo della Riccia: *"An Application of the Hopfield Model to Huffman Codes"*, *IEEE Trans. Inform. Theory*, vol. 39, no. 3, Mag. 1993, pp. 1071-1076.

[25] R. C. Merkele, M. E. Hellman: *"Hiding information and signatures in trapdoor knapsacks"*, *IEEE Trans. Inform. Theory*, vol. 24, n. 5, Set. 1978, pp. 525-530.

[26] Andrea Sgarro: *"Crittografia"*, Franco Muzzio Editore, Padova, 1993.

[27] S. Abe: *"Global Convergence and Suppressing of Spurious States of the Hopfield Neulal Networks"*, *IEEE Trans. Circuits Syst.*, vol. 40, no. 4, 1993, pp. 246-257.

[28] S. V. B. Aiyer, M. Niranjan, F. Fallside: *"A Theoretical Investigation into the Performance of the Hopfield Model"*, *IEEE Trans. on Neural Networks*, vol. 1, no. 2, Giu. 1990, pp. 204-215.

[29] D. Ingman, Y. Merlis: *"Local Minimum Escape Using Thermodynamic Properties of Neural Networks"*, *Neural Networks*, vol. 4, 1991, pp. 395-404.

[30] D. H. Ackley, G. E. Hinton, T. J. Sejnowski: *"A learning algorithm for Boltzmann Machines"*, *Cognitive Science*, vol. 9, pp. 147-169, 1985. Ristampato in *"Neurocomputing"* a cura di J. A. Anderson e E. Rosenfeld, MIT Press, Cambridge, 1988.

[31] AA. VV.: *"SIMULATED ANNEALING. Parallelization Techniques"*, Ed. R. Azencott, Wiley-Interscience Publication, USA, 1992.

[32] M. Peng, N. K. Gupta, A. F. Armitage: *"An Investigation into the Improvement of Local Minima of the Hopfield Network"*, *Neural Networks*, vol. 9, no. 7, 1996, pp. 1241-1253.

[33] K.W.Cheng, Tong Lee: *"Neural Network for Global Optimization"*, *RNNS/IEEE Symposium on Neuroinformatics and Neurocomputers*, Rostov on Don, Russia, October 7-10, 1992, pp. 53-63.

[34] Francesco Fabris: *comunicazione personale.*

[35] Antonella Mereu: *"Un'applicazione del modello di Hopfield alla crittografia"*, Tesi di Laurea in Scienze dell'Informazione, Facoltà di SS. MM. FF. NN., Università di Udine, A.A. 1993-94, Relatore Prof. Giacomo della Riccia.

[36] Peter Lancaster: *"Theory of Matrices"*, Academic Press, New York, 1969, p. 236.

Indice analitico