UNIVERSITÀ DEGLI STUDI DI ROMA "TOR VERGATA"

FACOLTA' DI INGEGNERIA

DOTTORATO DI RICERCA IN
Ingegneria dei sistemi sensoriali e di apprendimento

XXV CICLO

Titolo della tesi:

Asynchronous Spiking Neural Networks: paradigma generale e applicazioni.

Dottorando:

Gianluca Susi

A.A. 2011/2012

Tutor: **Prof. Mario Salerno**

Coordinatore: **Prof. Corrado Di Natale**

"L'*originalità* consiste nel tornare alle *origini*"

Antoni Gaudí

Sommario

Nelle Spiking Neural Networks (SNNs) l'attività neurale è rappresentata da *spike events* generati da *firing neurons* [1],[2],[3]. Il problema di base nella realizzazione di SNNs realistiche, concerne i tempi di arrivo apparentemente casuali dei segnali sinaptici [4],[5],[6]. In letteratura tecnica sono stati proposti molti metodi al fine di de-sincronizzare opportunamente le sequenze di spikes prodotte; tra le soluzioni biologicamente plausibili alcuni di essi considerano i tempi di transito lungo assoni o sinapsi [7],[8], mentre un differente approccio introduce il fenomeno della *spike latency* [9], proprietà del neurone dipendente dalle dinamiche interne. Prendendo in considerazione gli effetti legati allo stato del neurone, infatti, conseguentemente al raggiungimento di condizioni opportune, la effettiva generazione dello spike non avviene istantaneamente, ma dopo un certo intervallo, parecchio variabile nei differenti casi, perlopiù in una *immagine* continua.

SNNs che contemplano tali effetti, sono in grado di generare sequenze di spike con tempi di delay molto sensibili a piccole variazioni dello stato interno, variabili in insiemi resi continui dai fenomeni de-sincronizzanti. In questo lavoro viene presa in considerazione la *spike latency* come effetto de-sincronizzante significativo per la simulazione di SNNs realistiche.

Considerando ora un approccio *time-step based*, va da se che per realizzare simulazioni affidabili vengano richiesti intervalli di

campionamento molto piccoli. Al diminuire degli intervalli di campionamento, i processi di simulazione divengono molto piu lenti, a volte improponibili con la tecnologia oggi a nostra disposizione, ed il fenomeno parassita di *clusterizzazione* degli eventi, usando tale approccio, è comunque ineliminabile. Un approccio *event-driven based*, in questo scenario di *attività sparsa* e di alta *sensibilità a piccole variazioni*, è in grado di eliminare tali difficoltà, in quanto risulta molto più compatibile con la natura continua dei processi biologici, e la simulazione può essere agevolmente operata coinvolgendo sequenze di spikes molto più grandi [10],[11].

Tale classe di SNNs rappresenta un innovativo paradigma di simulazione in quanto il tempo è considerato come una variabile continua perchè è al più limitato dalla precisione ottenibile dal tipo di variabile implementata. Le proprietà che ne derivano conferiscono al sistema un gran numero di vantaggi che si traducono nella possibilità di simulare reti di grandi dimensioni in modo semplice, veloce e con una stretta fedeltà al caso reale.

Il presente lavoro di tesi introdurrà, nel primo capitolo, le reti neurali spiking come modelli di terza generazione, spiegando la differenza sostanziale con i precedenti; dopo una panoramica sulla relativa letteratura tecnica moderna, verranno evidenziati i punti di forza delle reti neurali di nuova realizzazione.

Il secondo capitolo avrà lo scopo di illustrare il modello di neurone implementato, in relazione a quelli classici piu importanti, evidenziando gli effetti presi in considerazione e spiegandone il perchè. Dopo aver compreso limiti e potenzialità delle diverse implementazioni già esistenti ed aver selezionato gli effetti da

riproporre nella unità elementare concepita, nel capitolo 3 si illustreranno le caratteristiche proprie del modello complessivo di rete realizzata, descrivendo il design dei vari livelli, come l'architettura di base e le connessioni sinaptiche, la plasticità e le metodologie implementative utilizzate, prendendo spunto per molti versi dal lavoro di G.Edelman *"Neural Darwinism. The Neuronal Group Selection theory"* [12].

Sfruttando i fenomeni esplorati, al fine di ottenere strumenti per la simulazione, nel capitolo 4 verrà illustrato il cuore della mia attività di ricerca: la sintesi di strutture funzionali, fondata su una logica sostenibile dal modello a disposizione e ispirata al comportamento delle reti neurali biologiche.

Con l'obiettivo di dimostrare la validità delle strutture come blocchi di processamento in applicazioni reali, nel capitolo 5 saranno descritte le tecniche di dimensionamento, *temporal coding based*, del sistema in rapporto alle funzionalità desiderate: design architetturale, *tuning* delle strutture elementari, composizione ed utilizzo delle stesse; verrà fatta una breve panoramica delle applicazioni già esistenti per questa classe di reti e contestualmente verrà affrontato un problema di classificazione con i gruppi sintetizzati, sfruttando le proprietà derivanti da un approccio *asynchronous SNN*-based, riportando i risultati ottenuti. Le ottime performance raggiunte dalla rete sia in termini di accuratezza nei risultati che nella velocità di esecuzione, ci lasciano chiaramente intendere che con tale metodo è possibile risolvere molti dei problemi finora affrontati con le reti neurali artificiali classiche, in maniera biologicamente molto più plausibile. La possibilità di sintetizzare gruppi funzionali dedicati di vario genere (detection, generazione di segnali, memorizzazione, classificazione,

etc.) ed interconnetterli all'interno della stessa rete, lascia spazio alla progettazione di sistemi atti allo svolgimento di task molto più complessi.

La direzione intrapresa, innovativa e finora poco esplorata in letteratura, è risultata premiante, infatti il nostro articolo *Spiking neural networks as analog dynamical systems: basic paradigm and simple applications* [13], presentato da me ad Amsterdam, nella conferenza internazionale *Advances in Computer Engineering* della ACEEE il giorno 7 giugno 2012, è stato onorato di riconoscimento con un *award* nella categoria *Best Paper*.

Indice

1. RETI NEURALI DI TERZA GENERAZIONE

Le reti neurali artificiali rappresentano essenzialmente un tentativo di riproduzione delle dinamiche computazionali che avvengono nelle dense reti di neuroni interconnessi componenti il sistema nervoso centrale nelle creature viventi. Queste consistono in sistemi di elaborazione elettronici distribuiti che cercano di processare informazioni tramite un numero generalmente grande di unità elementari, collegate tra di loro con una disciplina di una certa complessità. Nel caso reale, infatti, ogni neurone è mediamente collegato con una decina di migliaia di altri neuroni; si hanno quindi centinaia di miliardi di connessioni ed il comportamento intelligente emerge grazie alle numerose interazioni tra le unità interconnesse... ma come ha modo di realizzarsi una logica interna? E' possibile individuare strutture elementari in grado di assemblarsi ad un livello più alto e riuscire ad implementare funzioni più complesse?

In questa sezione verranno esposte le differenze tra le varie *generazioni* di reti neurali e si vedrà come, con un'interpretazione delle dinamiche interne delle reti biologiche in accordo con una branca della letteratura scientifica attuale, si arrivi ad includere effetti necessari per l'esistenza di alcune proprietà.

1.1 – Artificial Neural Networks classiche e Spiking Neural Networks

Nel 1943 W.S. McCulloch e Walter Pitts [14] proposero un modello di neurone consistente in un semplice combinatore lineare a soglia con dati binari multipli in entrata e un singolo dato binario in uscita.
Implementando questo modello di unità elementare, si svilupparono tutte quelle reti neurali artificiali oggi definite di *prima generazione* [1], adatte alla computazione con ingressi ed uscite digitali: con Rosenblatt nacque il primo schema di rete neurale [15], il Perceptron, e si svilupparono una varietà di modelli ad esso ispirati, come le reti di Hopfield o le macchine di Boltzmann, tutte con la caratteristica di poter emettere un output digitale. Di *seconda generazione* sono invece definite quelle reti comprendenti unità di calcolo che applicano una funzione di attivazione con un set continuo di possibili valori di uscita (generalmente funzioni *sigmoidali*), ad una somma pesata degli ingressi. A questa classe appartengono le reti *feedforward* e le *ricorrenti*, così come le *reti RBF*. Tali reti sono adatte al calcolo di funzioni con ingressi ed uscite analogiche. Un'altra loro importante caratteristica consiste nella possibilità di supportare algoritmi di apprendimento basati sul metodo del *gradient descent*, come la backpropagation.

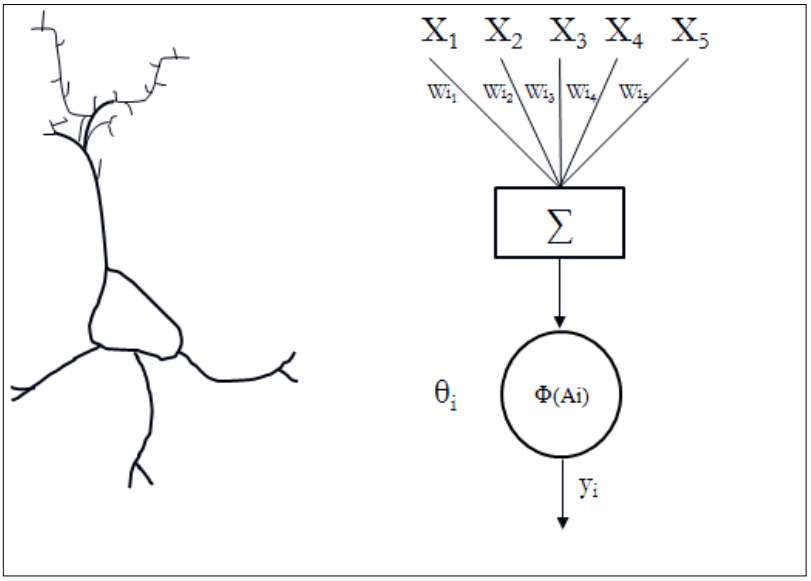


Fig.1: Neurone biologico (cellula piramidale) e sua schematizzazione formale.

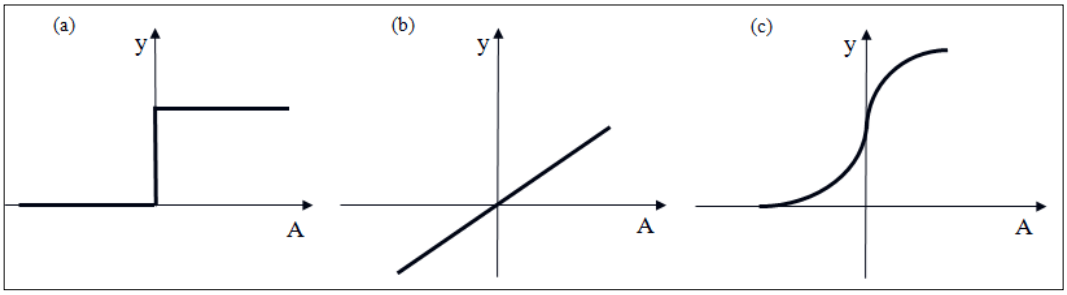


Fig.2: Funzioni di attivazione. Nella formulazione originale di McCulloch e Pitts la risposta del neurone è data da una funzione a gradino (a).

Per accostamento alla controparte biologica, gli output delle unità sigmoidali esibite dalle reti neurali di seconda generazione vengono interpretate come indicative del *firing rate corrente* di un neurone reale. Tali reti di seconda generazione sono, riguardo a questa interpretazione, biologicamente più realistiche rispetto ai modelli di prima generazione.

Evidenze sperimentali accumulate negli ultimi anni, hanno provato che molti sistemi neurali biologici usano il *timing* dei singoli potenziali d'azione (o "spikes") per codificare l'informazione.

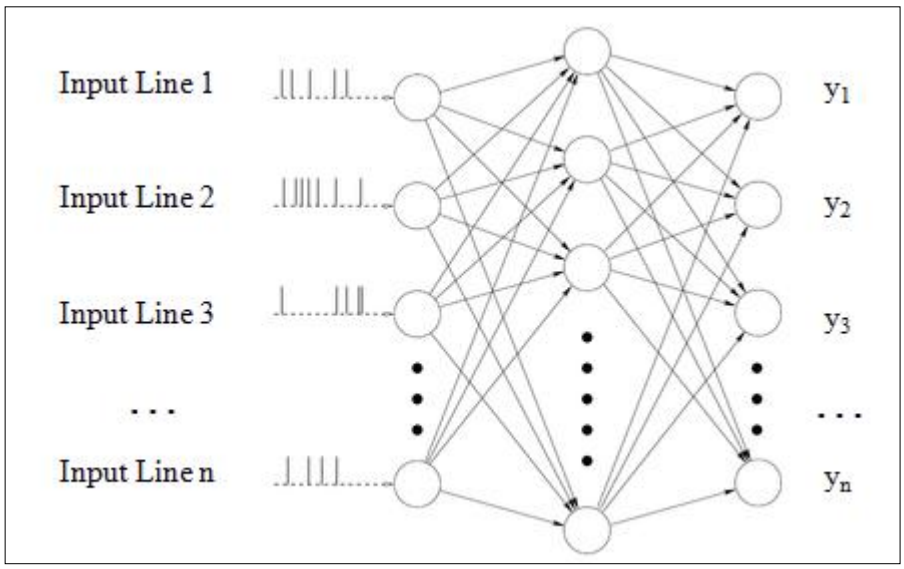


Fig.3: Schematizzazione di una spiking neural network. In questa classe di reti, viene usato il *timing* dei singoli potenziali d'azione per codificare l'informazione.

Tali risultati hanno dimostrato la inadeguatezza dei modelli fino a quel punto a disposizione per rappresentare una computazione biologicamente plausibile, suggerendo di investigare ulteriormente verso una *terza generazione* di reti neurali, oggi identificabile con quella delle *Spiking Neural Networks* [16].

1.2 – Spikes o rate?

Il titolo di questo paragrafo è ispirato dal dibattito sul significato che nel tempo si è cercato di attribuire alle modalità di trasmissione dell'informazione a livello neuronale. Prima di investigare su quale tra

i due sia il miglior meccanismo di codifica da utilizzare nelle reti neurali artificiali, verrà presentata una panoramica sui casi noti.

Un veloce sguardo alla letteratura sperimentale [16] rivela che in natura non esiste un unico e ben definito concetto di firing rate medio: infatti ci sono almeno tre differenti nozioni di *rate* che vengono spesso usate simultaneamente (o, addirittura, confuse):

- *Rate* inteso come *spike-count*, ovvero una media sul tempo. Tale concetto è usato non solo in esperimenti naturali, ma anche in modelli di reti neurali artificiali. Da questo punto di vista, gli spikes sono solo un modo conveniente di trasmettere l'output analogico in lunghe distanze: un treno di spikes con intervallo $\frac{1}{rate}$.

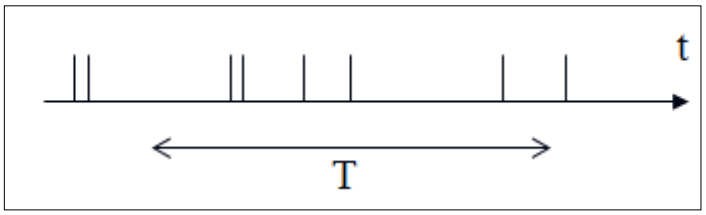


Fig.4: *Spike-count*: il rate è definito come media sul tempo (singolo neurone, singola sequenza)

Lo spike-count può essere calcolato come:

$$\upsilon = \frac{n_{sp}}{T} \tag{1}$$

In questo caso, il rate può dare un valore dopo solo 2 spikes, e le irregolarità incontrate in treni di impulsi reali devono essere considerate come rumore; ne emerge la necessità di effettuare la media su un gran numero di spike.

- *Rate* inteso come *spike-density*. Questa è una seconda definizione di rate, che necessita di più osservazioni per essere stimata. Per ogni piccolo intervallo di tempo $(t,\ t+\Delta t)$, lo sperimentatore conta il numero di volte che uno spike è avvenuto e somma i contributi su tutte le K ripetizioni dell'esperimento, riportando il risultato sull'istogramma *PSTH* (*Peri-Stimulus-Time-Histogram*).

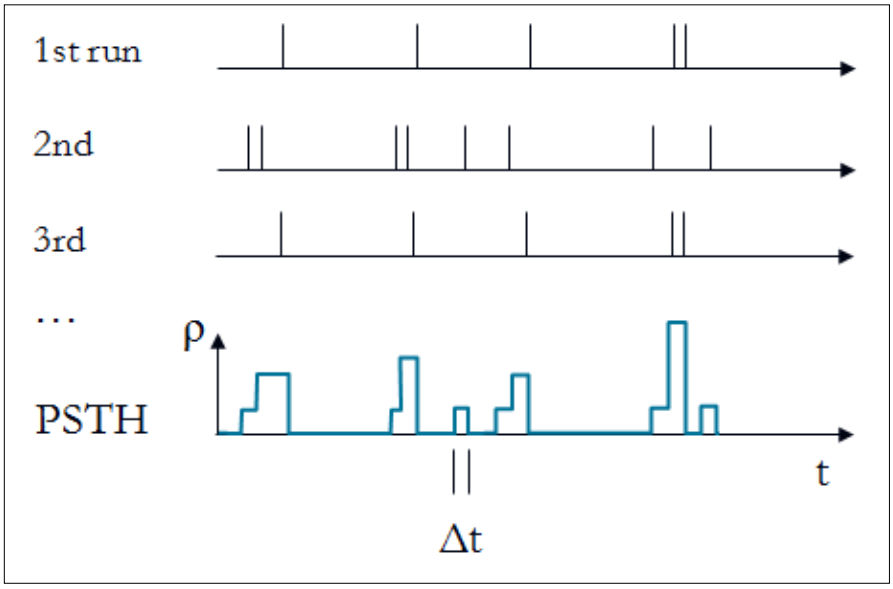


Fig.5: Definizione della densità di spike nel PSTH: il rate è considerato come media su un insieme di pattern

$$\rho(t) = \frac{1}{\Delta t}\frac{n(t;t+\Delta t)}{K} \tag{2}$$

Come procedura sperimentale, la misura della densità di spike è un utile metodo per valutare l'attività neuronale, ma tale approccio non può rappresentare lo schema di decodifica usato dai neuroni nel cervello. Ciò nonostante, le misure sperimentali di spike density possono avere senso se ci sono grandi popolazioni di neuroni che siano indipendenti l'una dall'altra e sensibili allo stesso stimolo.

- *Rate* inteso come *Population activity*. Supponendo che più popolazioni di neuroni abbiano proprietà simili (come nella *corteccia visiva primaria*), caratterizzate da stesso pattern di ingresso e medesime connessioni d'uscita, gli spikes di una popolazione *j* vengono inviati alla popolazione *k*.

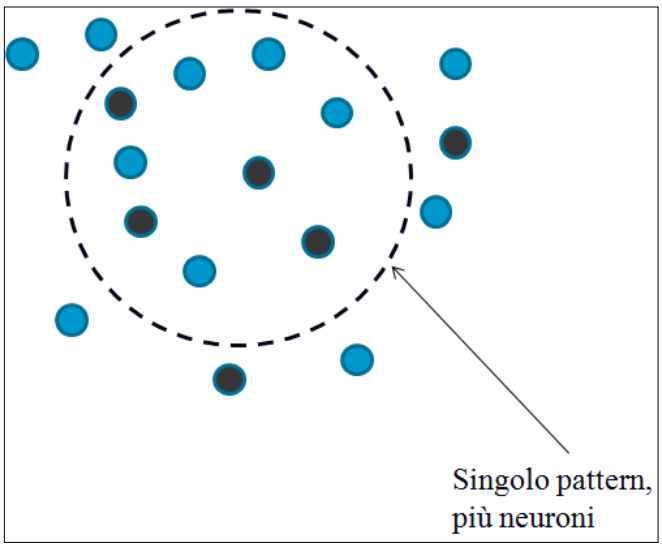


Fig.6: Definizione di poipulation activity

La quantità significativa, dal lato del neurone ricevente, è la porzione di neuroni attivi nella popolazione presinaptica j. Formalmente la *population activity* si definisce in questo modo:

$$A = \frac{1}{\Delta t} \frac{n_{act}(\Delta t)}{N} \tag{3}$$

Tale grandezza può variare rapidamente e riflettere cambiamenti nelle condizioni di stimolo quasi istantaneamente. Tale "*population activity*" non risente degli svantaggi di un firing rate definito da una media temporale a livello di singola unità. Il problema è che nella definizione stessa di tale grandezza, si ipotizza formalmente una

popolazione omogenea di neuroni con le stesse connessioni, cosa generalmente non realistica.

Alcune strategie di codifica dedotte dai modelli sono state confermate da evidenze sperimentali, ma la linea di divisione tra pulse codes e firing-rates non è mai stata così chiara come invece può sembrare: molti codici che sono stati prima proposti come esempi di pulse-codes, sono poi stati interpretati come variazioni di rate-codes. Dualmente, codici basati su spike timings si sono nel tempo scoperti compatibili con rate-codes. Senza entrare nel dettaglio di tali discussioni, approfondite in maniera esaustiva da molti testi [16], si deduce da un punto di vista funzionale l'importanza in tali codifiche di permettere ai neuroni di rispondere velocemente alle variazioni degli stimoli; si può facilmente intendere che codici che necessitano di finestre temporali troppo lunghe non sono utilizzabili per tale scopo.

Il concetto che l'informazione neurale fosse codificata nel firing rate dei neuroni è stato per molti anni il paradigma dominante in neurobiologia. Tale principio è infatti stato adottato per un lungo periodo nella teoria delle *artificial neural networks* di seconda generazione; recenti esperimenti in fisiologia hanno però dimostrato che in molte parti del sistema nervoso la codifica neurale è basata sul timing dei singoli potenziali d'azione, e la cosa è stata ampiamente confermata da riscontri su modelli matematici sviluppati. Tale scoperta ha fatto emergere una nuova classe di modelli neurali, le *spiking neural networks* o reti di terza generazione [1].

Reti composte da spiking neurons sono in grado di processare grandi quantità di dati usando un numero di spikes relativamente piccolo. Grazie alla loro similarità di funzionamento con i neuroni biologici, i modelli spiking sono in grado di offrire potenti tool per l'analisi di

alcuni processi cerebrali, come il processamento dell'informazione a livello neuronale, la plasticità e l'apprendimento. Allo stesso tempo, le reti neurali spiking offrono soluzioni in un gran range di specifici problemi ingegneristici, come signal processing veloce, event-detection, classificazione, riconoscimento vocale, e task come la navigazione spaziale o il controllo di motori [17],[25].

E' stato inoltre dimostrato che le SNN possono essere applicate non solo ai problemi risolvibili da reti neurali classiche, ma a situazioni di maggiore complessità in quanto, potenzialmente, molto più potenti di perceptrons e reti a funzione sigmoidale, in un'ottica computazionale. Per queste ragioni le SNN presentano un interesse scientifico in costante crescita.

Un problema insito sia nella generazione che nella rivelazione di sequenze di spikes consiste proprio nella sensibilità rispetto agli eventi ravvicinati, questione che nel seguente paragrafo verrà ampiamente presa in considerazione.

1.3 – Necessità di event-driven

Tradizionalmente la simulazione di reti neurali è basata su metodi time-step based, che hanno la caratteristica di sincronismo; in tali simulazioni, le variabili di stato vengono aggiornate ad ogni time step, in accordo con gli input correnti e i precedenti valori di tali variabili. Le espressioni differenziali descriventi le dinamiche neuronali del modello, sono generalmente calcolate con metodi di integrazione numerica, come quello di Eulero o Runge-Kutta. La precisione della integrazione numerica di tali variabili dipende dalla discretizzazione

di tali time steps: maggiore è la precisione che si vuole ottenere, più sono richiesti intervalli brevi, andando incontro a inconvenienti di consumo di tempo e potenza; peraltro, il fenomeno di sincronizzazione parassita è ineliminabile. Inoltre, per osservare interessanti fenomeni di gruppo, è necessaria la simulazione di grandi popolazioni con un'adeguata precisione e modelli dettagliati.

Eugene Izhikevich, uno dei più noti ricercatori contemporanei di SNNs, gia nel 2007 evidenzia suo malgrado le difficoltà emergenti in una simulazione di tipo *time-step based* su una rete composta da miliardi di neuroni descrivente una porzione di cervello umano: 1 secondo di attività simulata necessita di 50 giorni di simulazione, utilizzando 27 processori da 3 GHz. In una ottica ottimistica e facendo affidamento alla legge di Moore, l'autore afferma che le risorse computazionali necessarie per affrontare il problema saranno disponibili nei prossimi decenni.

REAL TIME ESTIMATION

Time	Number of processors	Processor speed
2006, January 1 (now)	116640000	3 GHz
2007, July 1	58320000	6 GHz
2009, January 1	29160000	12 GHz
2010, July 1	14580000	24 GHz
2012, January 1	7290000	48 GHz
2013, July 1	3645000	96 GHz
2015, January 1	1822500	192 GHz
2016, July 1	911250	384 GHz
...	...	...
...	...	...
2046, July 1	1	402653184 GHz

Fig.7: Previsione del numero di processori necessari alla simulazione, comparsa in un articolo di Eugene M. Izhikevich and Gerald M. Edelman del 2008.

I comuni simulatori Genesis e Neuron [18] sono utili per programmare precisi modelli biofisici di neuroni, con lo scopo di prevederne o

analizzarne il fedele comportamento, ma essi non sono efficienti per simulazioni veloci di SNNs di grandi dimensioni. Attualmente per tali scopi ci si basa prevalentemente su un approccio simulativo *time-driven*, che però diviene velocemente inefficiente al crescere delle dimensioni della rete.

In accordo con osservazioni di carattere biologico, i neuroni di una SNN sono connessi in maniera sparsa e irregolare nello spazio (*network topology),* e la grande variabilità del flusso di spikes implica che essi comunichino irregolarmente nel tempo. La bassa attività media, caratteristica delle SNN, rende un approccio time-step based inutile e "useless time-consuming" (Figura 8).

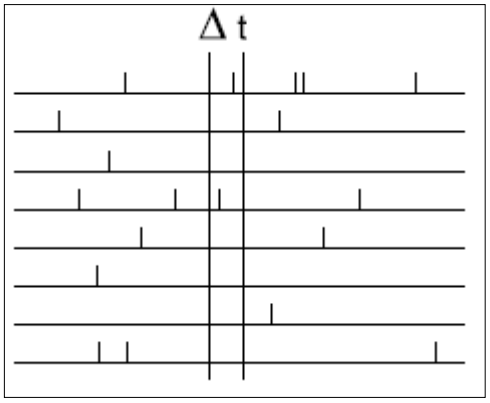


Fig.8: Fenomeno di sincronizzazione caratteristico della programmazione time-driven

Essendo l'interesse principale nella simulazione di SNN quello di prendere in considerazione il tempo esatto delle emissioni di spike, dunque la larghezza della finestra di tempo utilizzato per il calcolo discreto degli stati successivi di rete deve rimanere stretta, in modo che solo alcuni eventi spike possano verificarsi con lo stesso passo, anche in proporzione alle effettive connessioni esistenti. Dal momento che l'attività di una SNN può essere completamente descritta dalle emissioni di alcuni spikes da neuroni pre-sinaptici verso neuroni post-

sinaptici, un approccio di simulazione *event-driven* è chiaramente adatto per le simulazioni sequenziali di reti neurali spiking [19]. Più in generale, un approccio event-driven riduce sostanzialmente il carico computazionale di simulatori che controllano lo scambio di eventi, permettendo di non effettuare la scansione ad ogni passo temporale. In una tipica SNN, in qualsiasi momento, ogni neurone può ricevere sul proprio albero dendritico alcuni segnali emessi da altri neuroni.

Come già detto, molte comunicazioni nelle reti naturali sono effettuate per mezzo di spikes (potenziali d'azione) che sono eventi brevi e con un'alta sparsità nel tempo (mediamente poco frequenti); se l'evoluzione dello stato di un neurone tra tali spikes è deterministico, o la probabilità di tutti i target è nota, il numero di aggiornamenti di stato può quindi essere ridotto, accumulando il carico computazionale solo negli istanti in cui gli spikes vengono prodotti o ricevuti da un neurone.

2. NEURON MODELING

Un potenziale d'azione (o s*pike*) rappresenta un brusco e transitorio cambiamento del potenziale di membrana che si propaga agli altri neuroni tramite l'assone.

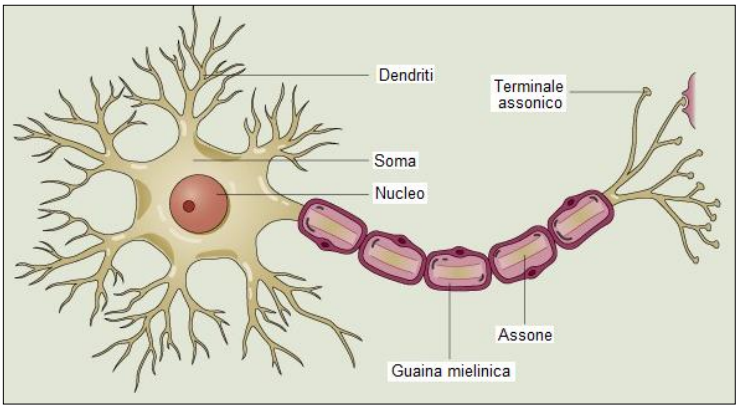


Fig.9: Rappresentazione schematica di un neurone.

Le connessioni tra unità avvengono grazie ai terminali sinaptici; presso tali siti, lo spike di uscita dei neuroni presinaptici stimola il rilascio di neurotrasmettitori, che a loro volta determinano, attraverso i recettori postsinaptici, un cambiamento nel potenziale di membrana dei neuroni postsinaptici.

Tali spike rappresentano il mezzo principale di comunicazione tra neuroni. La generazione di uno spike (detta *firing*) è generalmente il risultato di più impulsi che arrivano da altri neuroni: all'interno del neurone avviene il processo di *sommazione*. Per introdurre il discorso e facilitare la comprensione, può essere immaginato all'interno del neurone un valore di tensione di soglia (*firing threshold*). Se tale valore non viene raggiunto dai contributi in arrivo, il neurone rimane

inerte, altrimenti genera uno spike, avente la caratteristica *all-or-none.* Dopo tale evento, il potenziale di membrana attraversa un periodo di insensibilità agli stimoli (*periodo refrettario*), tornando gradualmente al *potenziale di riposo,* riacquistando man mano la sensibilità originale [20].

In figura viene mostrato il tipico andamento di un potenziale d'azione in funzione del tempo.

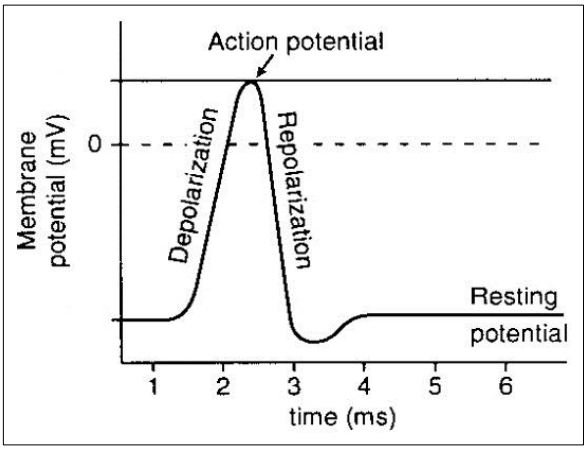


Fig.10: Andamento del potenziale d'azione.

Si osserva una prima fase crescente (*upstroke*) caratterizzata da una rapida depolarizzazione della membrana; questo cambiamento nel membrane potential continua fino a che non raggiunge un valore di picco (generalmente di circa 40 mV). A questa fase segue quella decrescente (*downstroke*) del potenziale d'azione, cioè una rapida ripolarizzazione fino a quando la membrana diventa più negativa del resting potential. L'ultima parte della fase decrescente viene chiamata iperpolarizzazione. Infine, c'è un graduale ristabilirsi del resting potential. Dall'inizio alla fine, il potenziale d'azione dura tipicamente circa 2 ms.

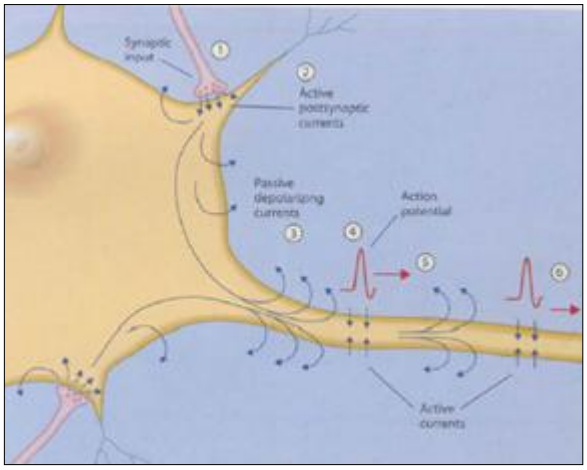


Fig.11: Propagazione dello spike lungo l'assone.

Il cambiamento nel potenziale di membrana da parte del neurone postsinaptico può essere negativo (per ingressi inibitori in quanto tali ingressi inibiscono il neurone postsinaptico dallo spike), o positivo (per ingressi eccitatori, in quanto tali ingressi eccitano il neurone postsinaptico alla generazione di uno spike). La grandezza del potenziale postsinaptico dipende dalla forza del terminale sinaptico.

Gli impulsi d'ingresso, quindi, comportano modifiche del potenziale di membrana in quanto effettivamente svolgono l'integrazione dei potenziali postsinaptici dal neurone. Il contributo di un potenziale postsinaptico è limitato nel tempo e decade con un'opportuna costante di tempo.

Diversi tipi di neuroni possono essere considerati in natura, con proprietà peculiari. D'altra parte, molti modelli sono stati introdotti e confrontati in termini di plausibilità biologica e costo computazionale. Agli antipodi vi sono il modello *Integrate and Fire* (derivato dal modello di L. Lapicque del 1907) e il modello di *Hodgkin-Huxley* (del 1952) il primo caratterizzato da un basso costo computazionale e

bassa fedeltà, mentre il secondo è una rappresentazione molto completa del caso reale.

Da un punto di vista elettrochimico, il neurone può essere caratterizzato dal suo potenziale di membrana V_m ; si illustreranno in seguito i due modelli più importanti, riportando gli aspetti principali di ognuno.

2.1 – Il modello Integrate & Fire

In questo modello (figura 12), semplice e largamente usato, si suppone che il neurone si comporti come un integratore dei segnali di ingresso, tutt'al più caratterizzato da una perdita intrinseca; non appena il potenziale di membrana supera un valore di soglia, il neurone emette uno spike, dopo il quale il potenziale ritorna al valore di riposo [5],[6].

Sviluppato da L. Lapicque nel 1907, il modello è formato da una capacità di membrana con una eventuale resistenza di leakage in parallelo. In tale modello la capacità viene caricata fin quando non viene raggiunta una certa soglia (*threshold*), cui segue una scarica che produce il potenziale d'azione (lo spike).

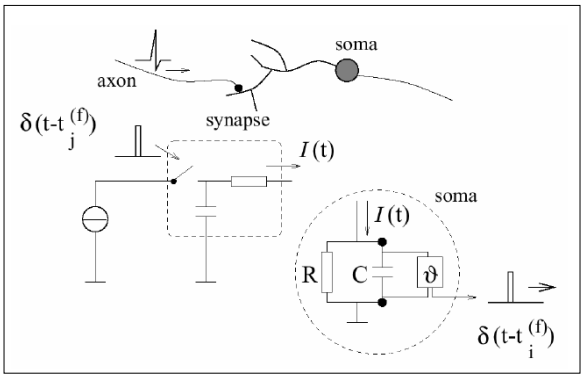


Fig.12: Schematizzazione del neurone Integrate & Fire

Il neurone in questo modello è con perdita (*leaky)*, in quanto la somma dei contributi del potenziale di membrana decade con una costante di tempo caratteristica (costante di tempo di membrana); trascurando tale decadimento su tutto l'intervallo di tempo si ottiene un integratore perfetto.

Una volta che la soglia prefissata viene raggiunta dal potenziale di membrana, si genera lo spike di uscita; il potenziale di membrana si riporta quindi al valore di riposo (*resting potential*). La legge di funzionamento del modello è:

$$C_m \frac{dv(t)}{dt} = I_{leak}(t) + I_s(t) + I_{inj}(t) \tag{4}$$

dove:

- $I_s(t)$ tiene conto dell'effetto dell'input sinaptico sul neurone;
- $I_{inj}(t)$ indica la corrente iniettata da un elettrodo intracellulare nel neurone [6];
- $I_{leak}(t)$ è il termine di corrente dovuto alle perdite della membrana:

$$I_{leak}(t) = -\frac{C_m}{\tau_m}[v(t) - V_0] \tag{5}$$

in cui V_0 è il resting potential e τ_m la costante di tempo di membrana passiva:

$$\tau_m = R_m C_m \tag{6}$$

R_m e C_m rappresentano rispettivamente resistenza e capacità di membrana e vengono assunte costanti.

Il modello presentato ha il vantaggio di poter essere simulato con una notevole velocità, ma naturalmente la sua semplicità lo rende poco

accurato dal punto di vista biologico, in quanto in grado di descrivere solo il fenomeno di spiking.

2.2 – Il modello Hodgkin-Huxley

Con una serie di articoli pubblicati a partire dal 1952, A.C. Hodgkin e A.F. Huxley hanno aperto le porte ad una comprensione dettagliata di come i segnali elettrofisiologici sono trasmessi all'interno del sistema nervoso. Proprio da questi lavori è nata la descrizione del potenziale d'azione [21].

Il modello di Hodgkin e Huxley esprime il comportamento elettrico di una cellula nervosa, descrivendo la corrente totale I di membrana come funzione del tempo e della tensione V di depolarizzazione di membrana. Definendo:

- E_r il valore assoluto del potenziale di riposo,
- V_{Na}, V_L e V_K le cadute di tensione su ciascun canale, (misura diretta dello scostamento rispetto al potenziale di riposo),
- C_m la capacità di membrana per unità di area,

si ha:

$$I = C_m \frac{dV}{dt} + g_K(V - V_K) + g_{Na}(V - V_{Na}) + \overline{g}_L(V - V_L) \tag{7}$$

in cui:

$$V = E - E_r; \tag{8}$$

$$V_{Na} = E_{Na} - E_r; \tag{9}$$

$$V_L = E_L - E_r \tag{10}$$

Contrariamente a quanto alcuni credono, Hodgkin e Huxley non vinsero il premio Nobel nel 1963 per aver introdotto il circuito equivalente della membrana poichè questa idea era già usata in elettrochimica, molto prima del 1952; un loro primo merito fu quello di introdurre nel circuito equivalente la dipendenza esplicita dal potenziale V, potenziale di scostamento da quello di riposo E_r della membrana o di ogni potenziale ionico (V_K , V_{Na} , V_{Cl}) come illustrato in figura 13. Si rimanda il lettore a testi specifici per un trattazione esauriente [21].

Nonostante il modello di Hodgkin-Huxley rimanga il più significativo e robusto modello matematico di cui si dispone nel campo delle neuroscienze, essendo basato su un sistema di equazioni differenziali interdipendenti è eccessivamente complicato e non risulta utile per simulare il comportamento di insiemi piuttosto numerosi di neuroni; questo non solo per l'impossibilità di trovare una soluzione analitica ma anche per il costo computazionale estremamente elevato in una elaborazione numerica.

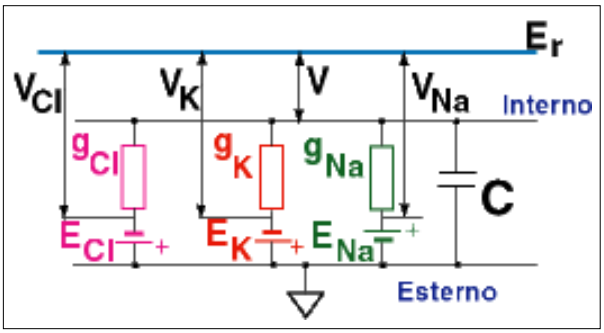


Fig.13: Circuito equivalente di una porzione di membrana in base al modello Hodgkin-Huxley. Nella figura, g_K , g_{Na} , g_{Cl} rappresentano le conduttanze (funzioni del tempo e del potenziale di membrana) dei canali a corrente ionica di potassio, di sodio e a corrente di perdita (leakage current, dovuta principalmente al cloro).

2.3 – Il modello realizzato

In vista della simulazione di reti neurali di grandi dimensioni, determinando i fenomeni chiave della trasmissione neurale è stato realizzato un modello in grado di contemplare i fenomeni fondamentali delle dinamiche neuronali. Come spiegato nell'introduzione del capitolo, il potenziale di membrana può variare grazie all'integrazione delle eccitazioni di ingresso. Dal momento che i contributi provenienti dall'esterno convergono costantemente in sommazione all'interno del neurone, un notevole accumulo di eccitazioni può portare il neurone al superamento di una soglia, provocando la generazione di uno spike in uscita.

Tuttavia, lo spike generato viene realizzato immediatamente, ma dopo un caratteristico tempo di ritardo chiamato *latenza*. Così, la *latenza* è il tempo che intercorre tra il superamento della soglia del potenziale di membrana e la generazione effettiva dello spike. Tale fenomeno, molte volte preso in considerazione nella letteratura tecnica è stato in questo lavoro ritenuto fondamentale per la modellizzazione del comportamento del neurone.

2.3.1 – Caratterizzazione della latenza

E' del tutto evidente che il concetto di *spike latency* (da qui in poi indicato semplicemente con *latency*) si basa su una definizione esatta del livello di soglia [11]. Tuttavia, la vera soglia non è un valore costante, in quanto dipende dalle precedenti attività del neurone, come

mostrato dalle equazioni di Hodgkin-Huxley. Inoltre, pur supponendo condizioni identiche a livello neuronale, stimoli di poco differenti possono generare comportamente completamente diversi.

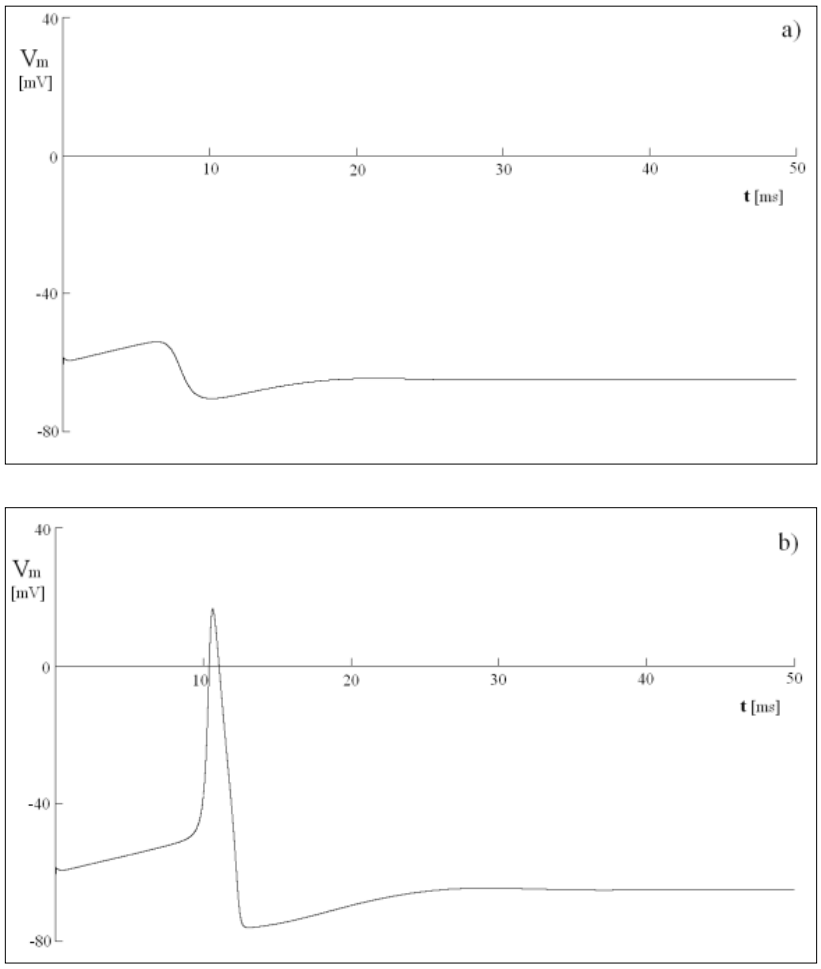


Fig.14: Andamento del potenziale di membrana causato da due impulsi di corrente di durata 0,01 ms applicati nelle stesse condizioni: potenziale di riposo a t=0. Le ampiezze dei due impulsi sono rispettivamente di *0.64523 nA* (figura a) e *0.64524 nA* (figura b). Gli andamenti, ottenuti dal simulatore *Neuron*, appaiono molto sensibili rispetto all'ampiezza degli stimoli forniti, evidenziando la caratteristica "All or None" di tali curve.

Il primo autore ad aver trattato il fenomeno la soglia da un punto di vista matematico è stato FitzHugh, il quale ha parlato per primo di Q*uasi Threshold Phenomenon* (*QTP*) [22]. Nel lavoro di FitzHugh viene definita una latenza massima, ma non vengono considerati nè una vera discontinuità nella transizione della risposta, né un esatto

valore di soglia. Infatti, con riferimento al modello dell'assone gigante del calamaro, è stato osservato che, a causa di fluttuazioni di membrana nel caso di osservazioni sperimentali, o di precisione insufficiente nel caso dei simulatori, non è possibile stabilire un valore preciso. Nelle figure14 (a) e (b), viene mostrata la risposta del neurone a due impulsi di corrente della stessa durata e di intensità leggermente differente, a partire dalle stesse condizioni iniziali; come si può osservare, il comportamento denota un'alta sensibilità rispetto a piccole variazioni della corrente di eccitazione.

Tuttavia, nel presente lavoro, è stato utilizzato un valore massimo di latenza ricavato dalle simulazioni effettuate, e applicato per stabilire una soglia di riferimento. Quando il potenziale di membrana diventa maggiore della soglia, la latenza appare come una funzione di V_m. Per ricavare tale valore e comprendere in maniera accurata il fenomeno della latenza, sono state effettuate opportune simulazioni su singoli neuroni.

Un altro fenomeno chiave della computazione neuronale inserito nel modello è il *decadimento sottosoglia*, che rende necessaria l'azione contemporanea degli spike afferenti affinchè il target superi la soglia. Tale fenomeno, in questo lavoro viene ritenuto lineare.

Per queste due caratteristiche il modello da noi realizzato può essere definito *leaky*, con *latency (LIFL: Leaky Integrate and Fire with Latency)*; per contemplare un comportamento più naturale possibile, è stato anche reso attivabile il *periodo refrattario*.

2.3.1.1 – Simulazioni sul singolo neurone

La *latency*, come detto, risulta strettamente legata al *firing threshold*, cioè al valore che deve assumere il membrane potential, affinché il neurone generi uno spike. Nel modello più semplice di neurone, l'Integrate & Fire, tale firing threshold è ben definita e costante ma, per come è costruito il modello matematico, non implica alcuna latency, quindi, ciò comporta un firing istantaneo che conseguentemente contraddice l'evidenza biologica [5],[6]. Per effettuare una caratterizzazione accurata della latency è stato necessario studiare un modello che tenga conto in maniera precisa dei meccanismi biologici, come quello descritto dalle equazioni di Hodgkin-Huxley.

In questa sezione verranno mostrate le proprietà significative della latenza, ricavate usando l'ambiente di simulazione NEURON [18], un tool per lo sviluppo veloce di modelli neuronali realistici, sulla base delle equazioni di Hodgkin-Huxley.

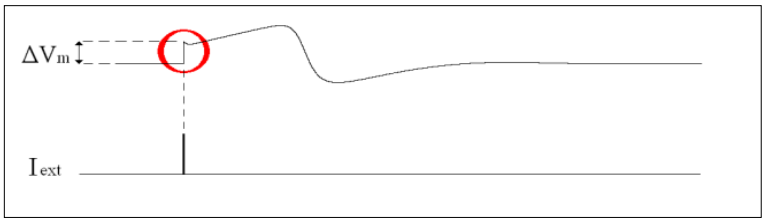


Fig.15: Variazione istantanea di ΔV_m per un impulso di corrente fornito I_{ext}. Caso particolare in cui l'eccitazione non è sufficiente a generare spike in uscita.

Il neurone può essere portato soprasoglia tramite l'iniezione di impulsi di corrente di ingresso, ricevuti da spikes generati dai neuroni afferenti. Poiché per impulsi di corrente di durata molto breve le variazioni istantanee del potenziale di membrana, (ΔV_m) presentano

una relazione di proporzionalità diretta rispetto agli impulsi di corrente connessi, la latenza può essere rappresentata come una funzione del potenziale di membrana. Si noti che, partendo dallo stato di riposo (V_{rest} = -65 mV), un impulso eccitatore di 1 nA di durata 0,01 ms, genera un ΔV_m = 10 mV.
Sono stati simulati a tal proposito tre casi significativi, utilizzando il passo di integrazione di 0,00125 ms e corrente di durata 0,01 ms.

α) Singolo impulso di corrente, eccitatore.

In questo caso, partendo dalla condizione di riposo, viene applicato un impulso di corrente di ingresso di ampiezza adeguata, in grado di causare spike di uscita dopo un certo tempo di latenza.
Simulando una serie di esempi, con impulsi di corrente nel range [0,64524 ÷ 10] nA, è stata deteminata curva della latenza, in funzione dell'ampiezza dell'impulso (o, equivalentemente, del potenziale di membrana V_m). Il comportamento della latenza è mostrato in figura 16, ove appare decrescente, con una forma quasi iperbolica. Il livello di soglia è risultato corrispondere ad una ampiezza di impulso uguale a 0,64524 nA, con una latenza (latenza massima) di 10,6313 ms. Impulsi con ampiezze inferiori, risultando sotto il livello di soglia, non sono in grado di generare spike d'uscita.

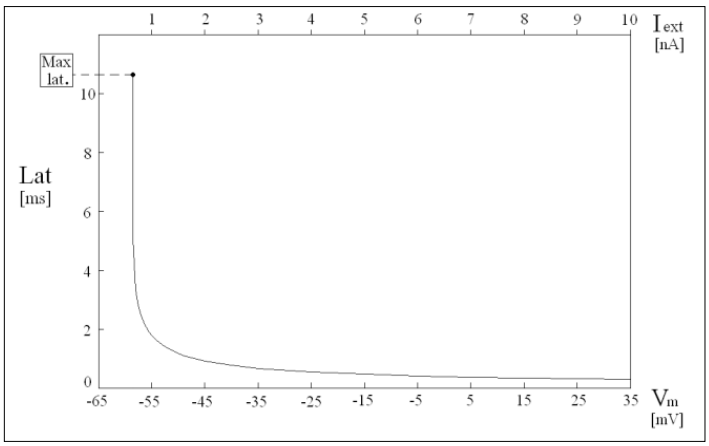


Fig.16. Latenza in funzione del potenziale di membrana (o, equivalentemente, della ampiezza di corrente *Iext*).

β **) Due impulsi di corrente, eccitatori.**

Partendo dalla condizione di riposo, dopo un primo impulso d'ingresso di corrente in grado di generare uno spike in uscita, è stato applicato un secondo impulso eccitatore per valutare la modifica dell'intervallo di latenza nel processo di spiking. Essendo il primo impulso di ampiezza fissa (pari a 0,64524 nA), la latenza complessiva è funzione dell'ampiezza del secondo impulso (variabile nel range [0,001 ÷ 5] nA) e del tempo di intervallo tra i due impulsi eccitatori di corrente.

In figura 17, viene mostrato il comportamento del tempo di latenza complessivo, cioè il tempo tra il primo impulso e il picco di uscita, in funzione dell'ampiezza del secondo impulso, per diversi valori di Δ. Si noti che la latenza generale diminuisce sempre all'aumentare dell'ampiezza secondo impulso; tuttavia, mentre l'effetto del secondo impulso è abbastanza rilevante per bassi valori di Δ, diventa meno importante quando Δ è alto, ossia avvicinandosi al valore della latenza corrispondente al primo impulso unico.

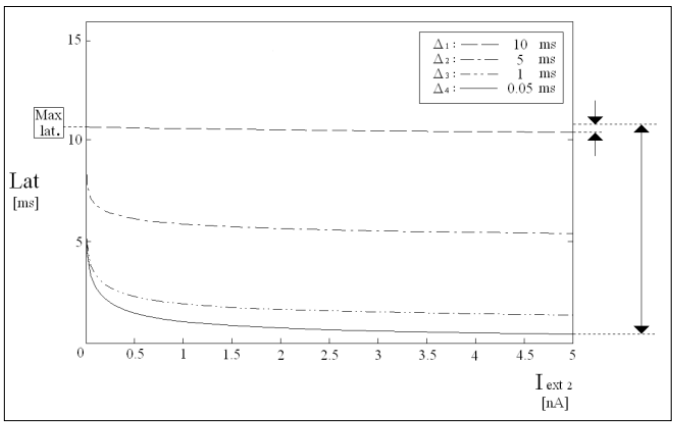


Fig.17. Insieme di curve corrispondenti a coppie di impulsi eccitatore-eccitatore. Ogni curva rappresenta la latenza generale in funzione dell'ampiezza del secondo impulso, per un intervallo di tempo Δ tra i due impulsi. L'effetto è maggiore per bassi valori di Δ.

γ) Primo impulso eccitatore e secondo impulso inibitore.

Partendo dalla condizione di riposo, dopo un primo impulso di corrente di ingresso in grado di causare un picco di uscita, un secondo impulso viene applicato durante l'intervallo di latenza, dopo l'intervallo di tempo Δ. Il secondo impulso è negativo, in modo da produrre un effetto inibitore. Mentre il primo impulso è di ampiezza fissa (pari a 3,0 nA, in grado di assicurare che il potenziale di membrana arrivi abbondantemente sopra la soglia), il secondo è di ampiezza variabile (partendo da -0,01 nA, fino ad un valore per il quale sia ancora possibile apprezzare lo spike di uscita).

Una volta applicato il secondo impulso, l'effetto principale consiste nell'incremento del ritardo nella generazione del potenziale di azione, causato dall'impulso inibitore. Più l'impulso inibitore è forte, maggiore è il ritardo provocato nella generazione del potenziale d'azione. Inoltre, più l'impulso inibitore agisce tempestivamente, maggiore è la latenza prodotta. Anche in questo caso, quindi, la latenza generale è

funzione dell'ampiezza del secondo impulso: analogamente al caso precedente, maggiore è l'intervallo tra i due impulsi, minore sarà l'influenza del secondo impulso (figura 18).

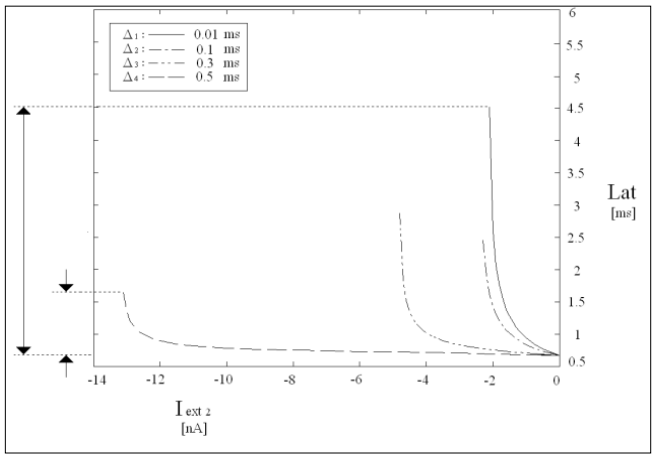


Fig. 18. Insieme di curve corrispondenti a coppie di impulsi eccitatore-inibitore. Ogni curva rappresenta la latenza generale in funzione dell'ampiezza del secondo impulso, per un intervallo di tempo Δ tra i due impulsi. L'effetto è maggiore per bassi valori di Δ.

Va notato che, qualora l'entità dell'impulso inibitore fosse molto forte o molto ravvicinata, si può verificare la totale inibizione dello spike.

2.3.1.2 – Modello neuronale semplificato in vista di reti neurali di grandi dimensioni

Con lo scopo di creare un simulatore di rete neurale che tenga conto del fenomeno della latency come effetto principale di desincronizzazione, il neurone è inteso come un sistema caratterizzato da una quantità reale non-negativa, lo *stato S*. Lo stato è modificato tramite la sommazione dinamica di segnali di ingresso P_i (chiamati *burning signals*) in arrivo al neurone in questione tramite gli input sinaptici. Tali segnali consistono nel prodotto tra i pesi presinaptici P_r

(parametro globale identificabile come relativo ai neuroni afferenti, o *spiking neuron*), ed i pesi postsinaptici P_w (quantità relative ai neuroni target, o *burning neuron*, e generalmente differenti tra loro). Definendo una *soglia normalizzata* $S_0 > 1$, possono essere identificate due differenti *modalità* di lavoro (figura 19):

a) per $S < S_0$, il neurone si dice in *modalità passiva*;

b) per $S > S_0$, il neurone si dice in *modalità attiva*.

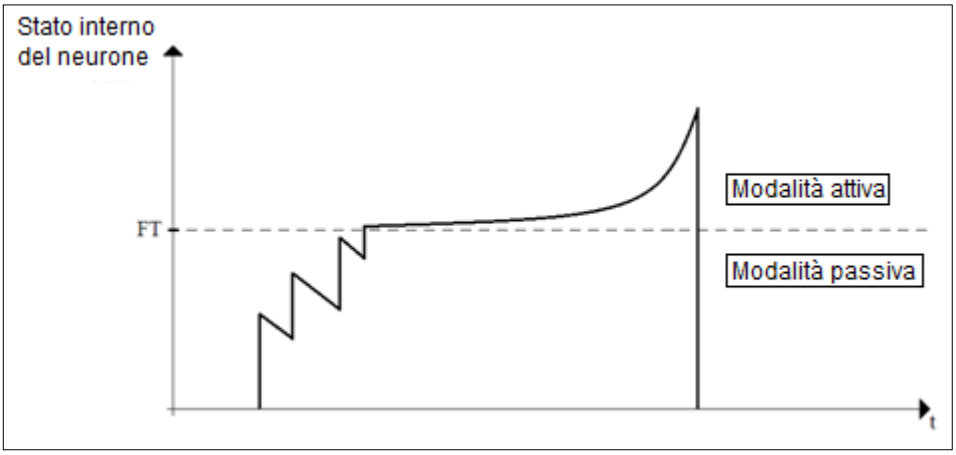


Fig.19: *Modalità* di lavoro *attiva* e *passiva*

In modalità passiva, un neurone può essere considerato un semplice accumulatore, come nell'approccio Integrate and Fire. Possono essere altresì considerati altri effetti significativi, come il *decadimento sottosoglia* (caratteristica *leaky*), per cui, in assenza di stimoli di ingresso, lo stato S non è costante col tempo ma tende a zero in maniera lineare. Per descrivere tale fenomeno si usa la formula:

$$S = S_A - \Delta t \, K_d \tag{11}$$

in cui S_A è lo stato precedente, Δt è l'intervallo di tempo e K_d è la *costante di decadimento.*

In modalità attiva, al fine di introdurre la *latency*, viene considerata una quantità reale positiva chiamata *time-to-fire* (t_f). Tale quantità rappresenta l'intervallo di tempo per cui un neurone attivo rimarrà ancora tale, prima che l'evento di *firing* sia effettivamente generato (figura 20).

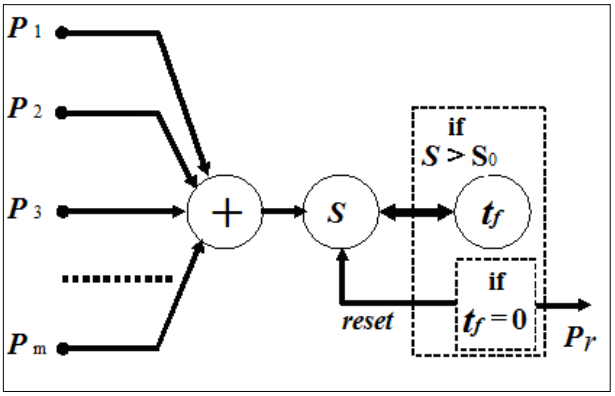


Fig.20: Modello proposto per il neurone

L'evento di firing è considerato istantaneo e consiste nei seguenti step:

a) generazione del segnale di output P_r, chiamato *peso presinaptico*, che viene trasmesso, attraverso le sinapsi d'uscita, ai neuroni riceventi, chiamati neuroni *burning*,

b) Azzeramento dello stato interno *S*, tale da far tornare il neurone nel suo stato di riposo,

Inoltre, nello stato attivo, si verifica una corrispondenza biunivoca tra t_f e lo stato interno *S* secondo la seguente relazione (*firing equation*):

$$t_f = \frac{1}{(S-1)} \tag{12}$$

per $S > S_0$. Il time-to-fire è in effetti una misura della *latency*, in quento esso rappresenta, come già detto, l'intervallo di tempo in cui il neurone rimane in stato attivo. Definendo il valore

$$S_0 = 1 + K_d \tag{13}$$

(con $K_d > 0$), il massimo valore del *time-to fire* è uguale a $1/K_d$. L'equazione del *time-to-fire* formalizza il comportamento analizzato in figura 21 ; analogamente al caso biologico, in cui la *latency* è una funzione del potenziale di membrana, il *time-to-fire* dipende dallo stato ***S*** descrivendo una iperbole equilatera.

Gli andamenti della curva derivante dalle simulazioni effettuate e quello della firing equation sono mostrati nella seguente figura.

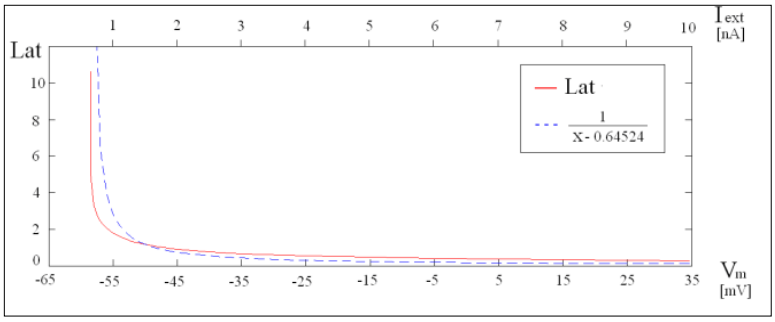


Fig.21: Paragone tra la curva della latenza e quello della firing equation. I due comportamenti sono molto simili.

Per i neuroni attivi vengono applicate le seguenti regole:

a) se sufficientemente stimolato, il neurone può entrare in modalità attiva, con un determinato stato S_A ed un corrispondente *time-to-fire* $t_{fA,}$ tale che:

$$t_{f_A} = 1/(S_A - 1) \tag{14}$$

b) In accordo con la biunivoca *firing equation*, quando il tempo cresce di t_x ($< t_{fA}$), il *time-to-fire* decresce al valore

$$t_{f_B} = t_{f_A} - t_x \tag{15}$$

e il corrispondente stato interno S_B raggiunge il valore

$$S_B = 1 + \frac{1}{t_{f_B}} \tag{16}$$

c) Se, dopo il tempo t_x viene ricevuto un nuovo impulso, nel calcolo dovrà essere considerato il nuovo stato interno S_B.
Come conseguenza diretta, l'effetto del nuovo impulso in ingresso diviene meno rilevante per valori di t_x più grandi (e quindi per valori di S_B più grandi).

La *firing equation* è il concetto fondamentale per rendere asincrona l'intera rete neurale: infatti, se questa relazione non fosse stata introdotta, il *time-to-fire* sarebbe rimasto uguale a zero e quindi ogni firing event sarebbe stato prodotto esattamente nell'istante di superamento del valore S_0 da parte dello stato *S*. Il *time-to-fire* rappresenta una quantità continua nel modello, rendendo l'evoluzione *asincrona* e il neurone dipendente dal modo in cui sia stata raggiunta la modalità attiva.
Per considerare segnali sia di tipo eccitatore che inibitore, nel modello sono contemplati pesi presinaptici sia positivi che negativi. Vengono considerati neuroni inibitori quelli che, possedendo $P_r<0$, sono in grado di abbassare lo stato interno *S* dei relativi burning neurons. Se un neurone in modalità attiva viene colpito dal potenziale d'azione prodotto da un neurone inibitore, il suo *time-to-fire* aumenterà; si può

inoltre verificare, in alcuni casi particolari, che il neurone colpito cambi la sua *modalità di funzionamento* da *attiva* a *passiva*, cancellando così la generazione di spike.
In conclusione, in presenza di potenziali d'azione provenienti sia da neuroni eccitatori che inibitori, si possono verificare le seguenti situazioni:

a. *Passive Burning*. In seguito ad un evento di burning, un neurone in modalità passiva non effettua una transizione alla modalità attiva in quanto il suo stato interno non supera la *firing threshold*.

b. *Passive to Active Burning*. Dopo l'evento di burning, un neurone in modalità passiva effettua una transizione alla modalità attiva in quanto il suo stato interno supera la soglia. Ciò è possibile solo in caso di eventi eccitatori. Un opportuno *time-to-fire* viene originato.

c. *Active Burning*. In seguito ad un evento di burning, un neurone in modalità attiva continua a rimanere nella stessa modalità in quanto il suo stato interno diviene minore della *firing threshold*.

d. *Active to Passive Burning*. Dopo l'evento di burning, un neurone in modalità attiva effettua una transizione alla modalità passiva in quanto il suo stato interno viene portato sotto la *firing threshold*. Ciò è possibile solo in caso di eventi inibitori. Il relativo *time-to-fire* viene cancellato.

Il programma di simulazione per reti neurali spiking tempo-continue realizzato, è basato su un metodo event-driven, in cui il tempo è una

quantità continua calcolata nel processo. Il programma prevede le seguenti fasi:

1. Definire la topologia di rete, ovvero creare collegamenti da neuroni *firing* a neuroni *burning* e relativi valori dei pesi postsinaptici.

2. Definire il valore da attribuire ai pesi presinaptici, che sarà lo stesso per ogni neurone sia eccitatore che inibitore della rete.

3. Scegliere dei valori iniziali per gli stati interni dei neuroni.

    3.1. Se tutti gli stati sono inferiori alla soglia S_0, non sono presenti neuroni attivi e quindi non è possibile ottenere attività nell'intera rete.

4. Calcolare il *time-to-fire* per tutti i neuroni attivi.
Applicare ora la seguente procedura di simulazione ciclica:

    4.1. Trovare il neurone N_0 con il minimo *time-to-fire*.

    4.2. Applicare un incremento temporale uguale a t_{f0}; aggiornare gli stati interni e il *time-to-fire* per tutti i neuroni attivi.

    4.3. Spiking del neurone N_0. Secondo questo evento, il neurone N_0 torna passivo e vengono aggiornati gli stati di tutti i neuroni ad esso direttamente collegati, secondo le regole prima descritte.

    4.4. Aggiornare il set di neuroni attivi. Se non sono presenti altri

neuroni attivi, la simulazione è terminata. Altrimenti, ripetere tutto il passaggio 4.

Al fine di tener conto della presenza di sorgenti esterne, alcune semplici modifiche sono state applicate alla procedura sopra descritta. In particolare, i segnali di ingresso sono considerati in maniera simile agli spike generati all'interno della rete, anche se dipendenti da eventi esterni. Le sequenze di spike di ingresso afferiscono ad alcuni specifici neuroni burning attraverso adeguate sinapsi esterne. Pertanto, l'insieme di neuroni di ingresso è fisso, e le sequenze di spike esterne vengono moltiplicate per gli opportuni pesi postsinaptici. È evidente che i parametri delle fonti esterne non sono in nessun modo condizionate dalla evoluzione interna della rete. Pertanto, quando nella procedura precedente viene considerata la lista dei neuroni attivi e il *time-to-fire*, il discorso si estende anche alle sorgenti esterne.

In precedenti lavori, noti in letteratura tecnica, il metodo sopra riportato è stato già applicato a reti neurali con circa 10^5 neuroni. L'insieme di pesi postsinaptici è stato continuamente aggiornato in base alle regole classiche di plasticità, ed interessanti effetti globali sono stati riscontrati, come la teoria della selezione dei gruppi neuronali, introdotto da Edelman.

Successivamente, il modello è stato applicato a piccole strutture neurali per realizzare determinati task; in questi casi, nessuna regola di plasticità è stata implementata, pertanto i valori di pesi presinaptici e postsinaptici del sistema sono stati considerati come dati costanti. Grazie alla presenza dell'effetto di latenza, che conferisce alla rete il carattere *asincrono*, le strutture proposte sembrano molto adatte per generare o rilevare sequenze di spike giacenti in range *tempo-continue*. Le strutture neurali discusse sono molto semplici e possono

essere facilmente connesse per arrivare ad applicazioni più complesse. Tutti gli schemi neurali proposti sono stati testati tramite la realizzazione di un simulatore in ambiente MATLAB.

3. NETWORK MODELING

Nel presente capitolo sarà descritto in maniera approfondita il modello di rete realizzato. Tale modello permette di generare reti di dimensione arbitraria, lasciando la possibilità di creare connessioni desidearate dall'utente. E' possibile lasciare la rete in evoluzione libera da una configurazione di partenza casuale o sintetizzare gruppi neurali in grado di implementare specifiche funzioni.

Come si vedrà in seguito, è possibile la connessione di gruppi neurali interni (figura 22) e sono visibili particolari effetti globali.

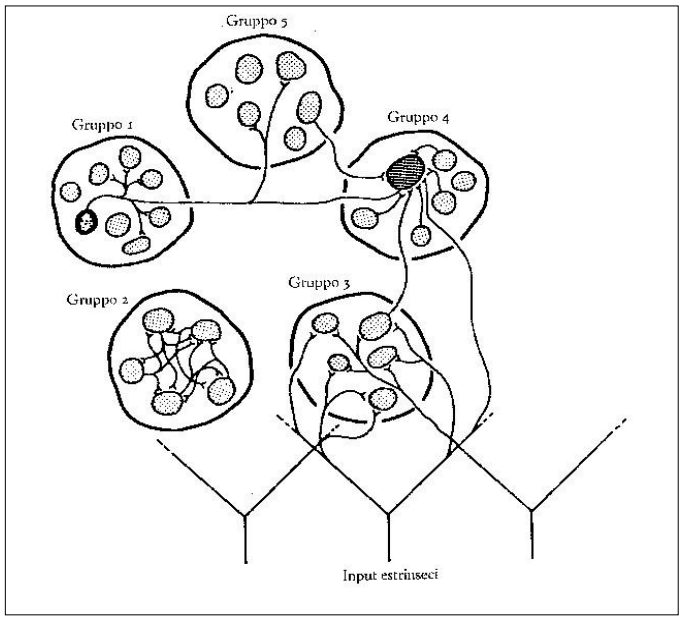


Fig.22: Connessione di gruppi funzionali nelle mappe neurali (Gerald M. Edelman, [12]).

3.1 – Il paradigma

La mappa di connessione dei neuroni viene definita stabilendo per ogni neurone *firing* i neuroni *burning* direttamente connessi tramite adeguate sinapsi, attraverso i pesi postsinaptici. E' evidente che la differenza tra i neuroni firing e burning è solo tesa a definire le sinapsi e la topologia di rete. Dunque, ogni neurone può essere pensato sia *firing* che *burning*, visto che può generare o ricevere un segnale spiking. L'intera distribuzione delle sinapsi può essere memorizzata in una generica matrice di pesi postsinaptici [P_w], di dimensioni $N \, x \, N$, ove N rappresenta il numero totale di neuroni. Ogni elemento della matrice rappresenta il peso postsinaptico relativo alla connessione da un neurone firing a burning. Se tale sinapsi non è presente, l'elemento è nullo; generalmente la mappa delle connessioni risulta altamente incompleta e la matrice risulta sparsa.

Inoltre, per grandi quantità di neuroni, la matrice completa $N \, x \, N$ può risultare non facilmente memorizzabile; pertanto, al fine di ottimizzare i requisiti di memoria, è stata applicata una tecnica di gestione di matrici sparse. Molte topologie di rete possono essere implementate con questa tecnica; nel programma di simulazione proposto vengono di default considerate architetture a connessione locale come nel caso delle *Cellular Neural Networks*. Ogni neurone firing è connesso direttamente ad un certo numero di neuroni burning appartenenti ad un vicinato di ordine stabilito. Le mappe di connessione locale sono mostrate in figure 23 e 24 che indicano gli schemi di connessione tra neuroni eccitatori (en) ed inibitori (in).

Nel simulatore sono definite le seguenti classi di sinapsi: s_{ee} , s_{ei} , $s_{ie,}$ mentre le sinapsi s_{ii} non sono mai presenti (il pedice *e* sta a rappresentare i neuroni eccitatori, mentre il pedice *i* quelli inibitori)

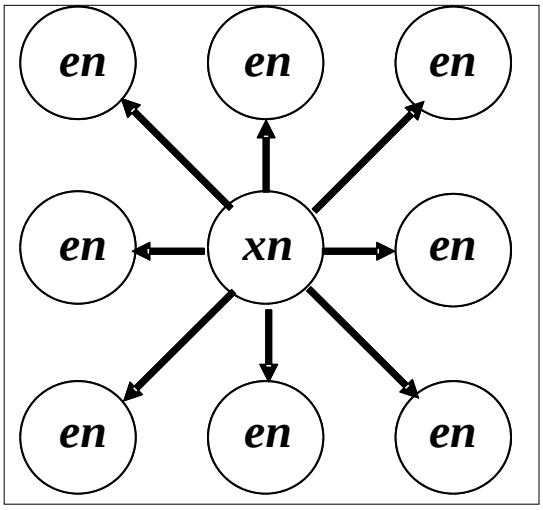


Fig.23: Mappa delle sinapsi s_{ee} e s_{ie} ; ***xn*** sta per neuroni firing eccitatori o inibitori, mentre ***en*** per neuroni burning eccitatori.

Come mostrato in letteratura tecnica, i neuroni inibitori coinvolgono vicinati più grandi.

Dunque, nel caso di sinapsi s_{ei} , i neuroni burning inibitori non sono confinanti con le relative controparti firing eccitatorie. Le figure 23 e 24 si riferiscono ai casi di vicinato minimo.

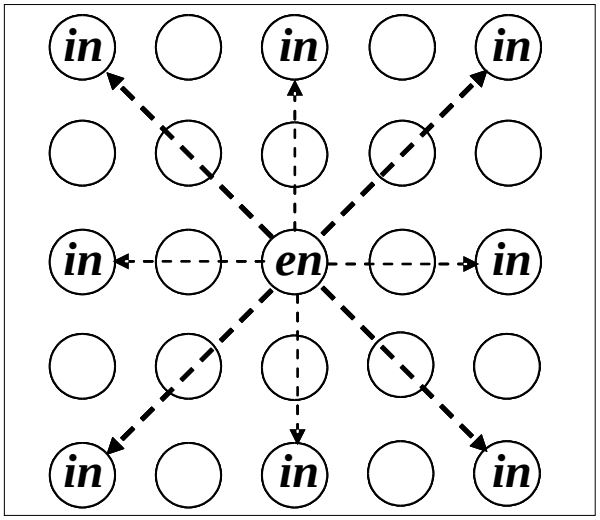


Fig.24: Mappa delle sinapsi s_{ei} ; ***en*** sta per neuroni firing eccitatori, mentre ***in*** per neuroni burning inibitori.

3.1.1 – Plasticity

La plasticità consiste in una certa variazione del peso postsinaptico P_w in accordo con le dinamiche neuronali. Tutti i pesi P_w sono confinati tra un valore minimo ed un massimo, e sono sempre quantità positive. La classica regola di Hebb è stato provato non essere idonea a modellare opportunamente il complesso comportamento plastico del reale sistema nervoso. Nel modello realizzato, sono stati implementati tre regole di plasticità, in accordo con Edelman.

- *Exponential decay*: tutti i pesi postsinaptici decrescono verso un valore minimo in maniera esponenziale, con una determinata costante di tempo.

- *Heterosynaptic enhancement:* quando viene determinato, da una certa sinapsi, un evento di burning, il peso postsinaptico viene

incrementato in funzione dei precedenti eventi di burning provenienti da altre sinapsi sullo stesso neurone, in una specifica finestra temporale (finestra eterosinaptica).

- *Homosynaptic enhancement:* quando viene determinato da una certa sinapsi un evento di burning, il peso postsinaptico viene incrementato in funzione dei precedenti eventi di burning provenienti dalla stessa sinapsi sullo stesso neurone, in una specifica finestra temporale (finestra omosinaptica).

Le costanti di aumento e decremento relative alle regole etero ed omosinaptiche possono essere adeguatamente dimensionate.

3.2 – Effetti globali

Siccome la generazione di spike riporta sempre il neurone firing in stato passivo, le frequenze dei burnings di classe a) e b) (precedentemente definite) devono essere sufficientemente alte. Burning di classe c) sono solo in grado di modificare l'evoluzione del tempo, e burning di classe d) riconoscono l'attività globale. Su queste basi molti criteri possono essere introdotti per valutare il livello di attività dell'intera rete.
Un parametro significativo per cui la stabilità globale può essere controllata è il peso presinaptico P_r che rappresenta l'ampiezza dello spike generato da ciascun neurone firing della mappa. Infatti, bassi valori di P_r sono in grado di portare alla riduzione del numero totale di neuroni firing, fino al termine di qualsiasi attività. Al contrario, valori

elevati di P_r sono in grado di far aumentare il numero di spike, fino alla saturazione del sistema. Un valore abbastanza corretto del P_r può essere scelto in funzione del numero di sinapsi a partire da un dato neurone firing (*fan-out*, f_o). Poiché ogni evento di firing produce l'azzeramento dello stato interno S da S_0 (soglia normalizzata) a 0, il valore di S_0 può essere pensato distribuito tra tutte le sinapsi in uscita dal neurone. Così, una scelta utile è

$$P_r = S_0 / f_o \tag{17}$$

Tale scelta si è dimostrata adatta per garantire la stabilità della rete.

I segnali di ingresso sono considerati come gli spike dei neuroni firing, anche se dipendenti da eventi esterni. Sequenze di spike di ingresso arrivano a neuroni specifici attraverso opportune sinapsi esterne. Gli spike delle sequenze in arrivo, durante il processo di burning, vengono moltiplicati per adeguati pesi postsinaptici. Anche questi pesi sono sottoposti alle regole descritte dalla plasticità, così gli stessi ingressi possono essere collegati a diversi neuroni burning, mostrando effetti diversi nei vari casi.

Sono stati effettuati diversi test di simulazione sul paradigma neurale proposto. In molti casi, come punto di partenza per la simulazione sono stati scelti valori casuali per i pesi postsinaptici. Simulando opportune sequenze di input, mantenendo attive le regole di plasticità proposte, si verificano alcune proprietà della evoluzione della rete, in particolare la selezione di specifici gruppi neurali, in cui compare attività ad un livello abbastanza elevato, mentre rimane bassa nelle regioni tra i diversi gruppi. Questa proprietà di auto-confinamento dell'attività rete sembra rimanere stabile, anche quando l'applicazione dell'ingresso considerato viene terminata.

Come esempio, si propone una rete neurale dalle seguenti caratteristiche:

Numero di neuroni eccitatori = 18060

Numero di neuroni inibitori = 2021

Numero di sorgenti esterne = 25

I neuroni eccitatori sono disposti in una metrice bidimensionale, di dimensione 140 x 129; Scegliendo un vicinato di ordine 4, viene generato un numero di pesi postsinaptici pari a 1608220; inizialmente i valori sono scelti in maniera casuale in un range prestabilito. Con l'obiettivo di generare una rete capace di mostrare una attività iniziale, vengono assegnati opportuni valori iniziali per gli stati S . In questo modo il numero iniziale di neuroni attivi è pari a 2452.

In figura 25 è rappresentata una mappa di rete, in cui ogni neurone è rappresentato da un punto. Punti più chiari rappresentano neuroni con un basso valore dello stato interno *S*, mentre gradazioni più scure indicano valori maggiori di *S*.

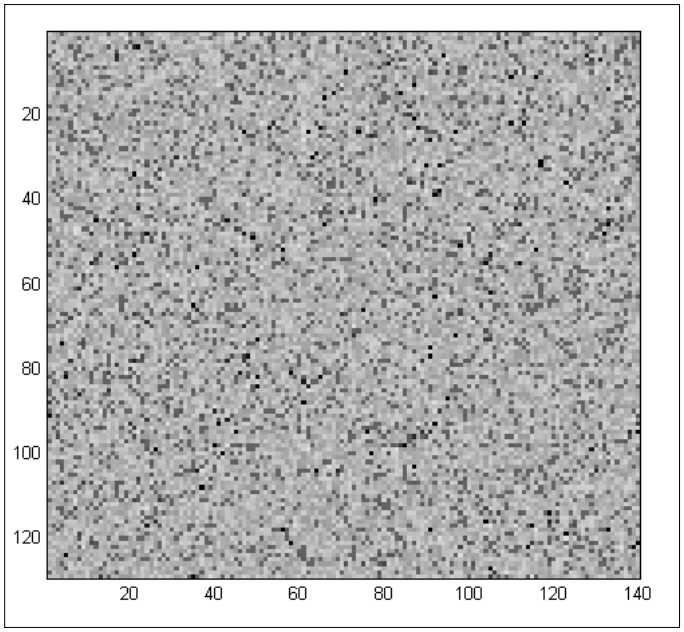


Fig.25: Mappa della rete neurale usata nella simulazione. Ogni punto rappresenta un singolo neurone. Punti più chiari rappresentano bassi valori di S, mentre punti più scuri sono associati a valori più alti. La mappa si riferisce alla configurazione iniziale, prima del processamento.

Vengono di seguito presentati i parametri di simulazione della rete dopo un numero di firing pari a 22518 [11]. Il *tempo normalizzato di simulazione* interno è uguale a 116.5, dove il numero di neuroni attivi è ridotto a 1096. La nuova mappa neuronale è mostrata in figura 26, in cui l'attività della rete non appare uniforme come quella mostrata in figura 25. Infatti, si potranno notare nella figura cinque gruppi specifici, in cui risulta attiva la maggior parte dei neuroni. Le frontiere dei gruppi sono ben delineate e l'attività nelle regioni tra di loro interposte è vicina a zero.

I parametri di simulazione coinvolti nel processo dalla mappa iniziale di figura 25 a quella di figura 26, sono i seguenti:

Numero di firing = 22517

Numero di burning:

passive = 1163500

passive-to-active = 21020

active = 568891

active-to-passive = 1223

I parametri coinvolti nelle regole di plasticità sono i seguenti:

Numero di etherosynaptic upgrading = 39469900

Numero di homosynaptic upgrading = 2341430

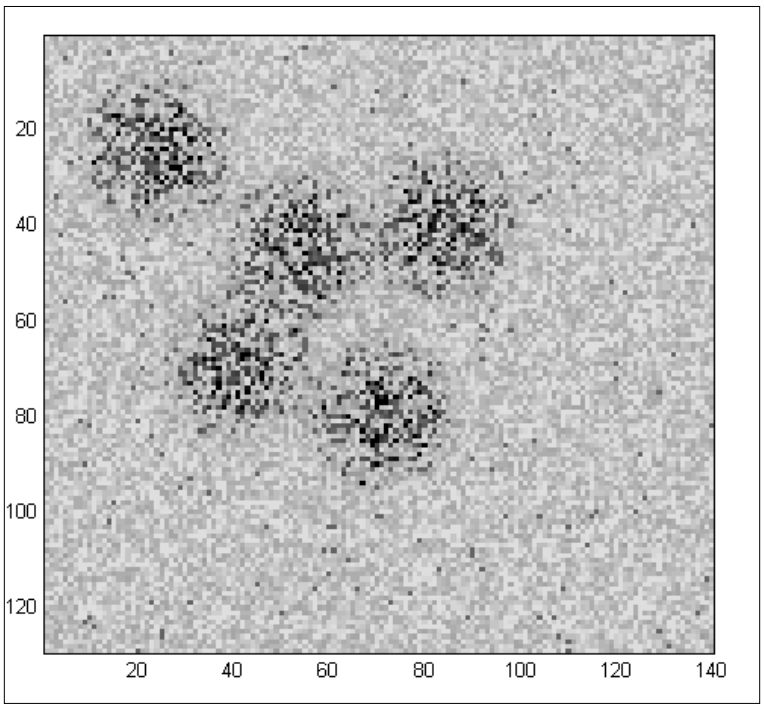


Fig.26: Situazione della mappa di figura 25, dopo la simulazione. La *Selezione dei Gruppi Neuronali* è chiaramente visibile.

La durata tipica riscontrata nelle performance nel test di simulazione è di circa 11 minuti, su un Pentium dual core 2.5 GHz (ram: 2GB).

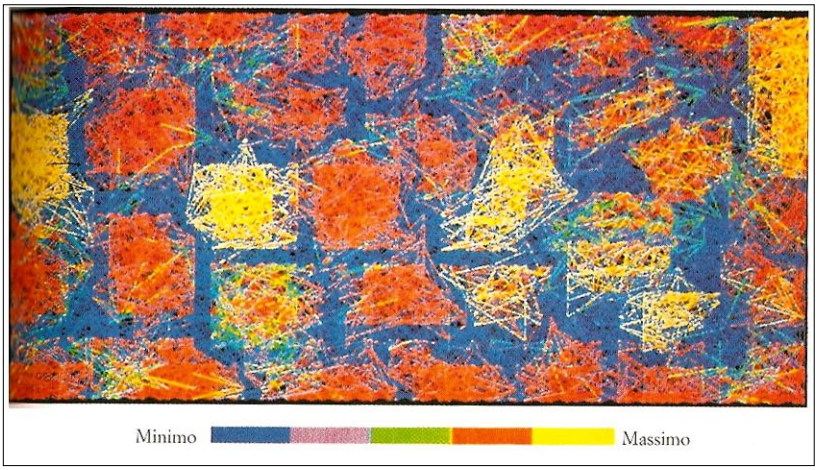


Fig.27: Formazione di gruppi neuronali ottenuta da una simulazione del 1987, su IBM 3090 da Pearson.

4. SINTESI DI GRUPPI FUNZIONALI

L'intenzione effettiva di questo lavoro di tesi è stata prevalentemente quella di trovare una logica, ispirata al funzionamento delle reti neurali biologiche, sostenibile in relazione al modello a disposizione. Infatti, come visto nei precedenti capitoli, gli effetti presi in considerazione nel modello realizzato derivano da un'attenta analisi sia a livello intrinseco del neurone (comportamento estrapolato da simulazioni su software *Neuron*), che a livello di rete (architettura e regole di funzionamento estrapolate da *"Neural Darwinism"* di Gerald Edelman) [12].

Sperimentando in questa direzione, è stato possibile sintetizzare gruppi funzionali di neuroni, applicabili per molteplici task come generazione di segnali, memorizzazione e classificazione.

Tale ricerca è stata operata combinando ad un livello superiore i semplici effetti ottenibili da alcune configurazioni di base, variandone connessioni e pesi ed evidenziandone gli esiti più interessanti. Comprovando volta per volta i risultati ottenuti con il modello matematico a disposizione, si è arrivati ad acquisire la padronanza di poter progettare strutture dedicate, a partire da specifiche su comportamento desiderato.

La direzione intrapresa, innovativa e praticamente inesplorata in letteratura, è risultata premiante, infatti l'articolo "*Spiking neural networks as analog dynamical systems: basic paradigm and simple applications"* (M.Salerno, G.Susi, A.Cristini, A.D'annessa,

Y.Sanfelice), da me presentato alla Conferenza Internazionale *Advances in Computer Engineering* della ACEEE il giorno 7 giugno 2012, è stato onorato di riconoscimento con un *award* nella categoria *best paper*.

Nel corso di questa sezione verrà illustrata la ricerca effettuata, spiegando via via motivando le direzioni intraprese; mostrando i risultati raggiunti verranno infine messe in evidenza le potenzialità di tali strutture.

4.1 – Definizioni e configurazioni di base

Nell'ottica descritta, la *temporal coding* acquista una importanza fondamentale nella trasmissione dell'informazione a livello neurale; è per questo che il lavoro svolto durante la mia attività si è rivolto proprio alla ricerca di strutture di semplice architettura che riuscissero, in maniera compatibile col modello realizzato, a svolgere determinate funzioni.

In questa sezione, la precedente teoria verrà applicata a piccoli gruppi di neuroni in modo che possano essere introdotte un certo numero di semplici applicazioni tempo-continue.

Un'attività logica molto semplice consiste nel riconoscimento di appropriate sequenze di spike di ingresso, tramite adeguate strutture neurali [23],[24]. Le sequenze di input differiscono dal timing di arrivo degli spike di cui sono composte, e il riconoscimento consiste nella generazione di spike da parte di un'unità di uscita, chiamata *Target Neuron (TN)*. Sarà dapprima introdotta una funzione di base

chiamata *sommazione dinamica*, fondamentale per la realizzazione di molte strutture semplici che sfruttano gli spike-timings alla base del proprio funzionamento.

La funzione di base di un *TN* è quella di produrre un picco di uscita quando un numero di spike viene ricevuto in un certo intervallo di tempo (figura 28). In un target opportunamente tarato su una certa attività, lo spike di uscita può non essere generato; infatti, risulta evidente che il neurone target diventa attivo quando le seguenti due condizioni sono contemporaneamente soddisfatte:

1. la somma delle ampiezze degli spike in arrivo è maggiore della soglia *FT*.
2. gli spike in ingresso sono opportunamente sincronizzati.

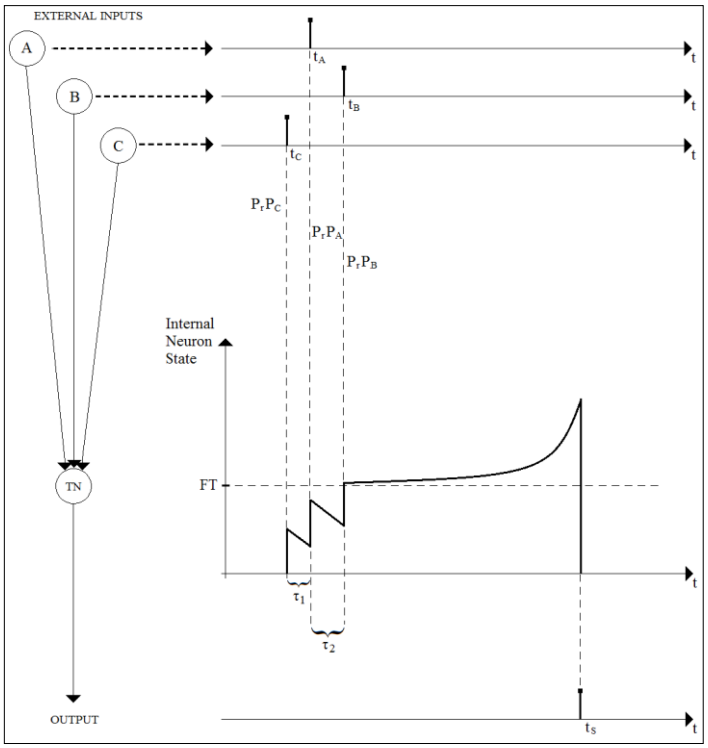


Fig.28: Tre spike vengono inviati ad un neurone target per generare uno spike di uscita; i tempi τ_1 e τ_2 rappresentano gli interspike intervals.. E' chiaramente visibile il ruolo che gioca il decadimento sottosoglia sulla necessità di sincronismo degli ingressi.

L'effetto del decadimento sottosoglia viene usato in questo meccanismo come parametro di progetto. Dato il suo effetto, risulta evidente che per tempi di arrivo meno sincronizzati sono necessari spike di ingresso con ampiezze maggiori: solo in questo modo potrà essere comunque raggiunta la soglia.

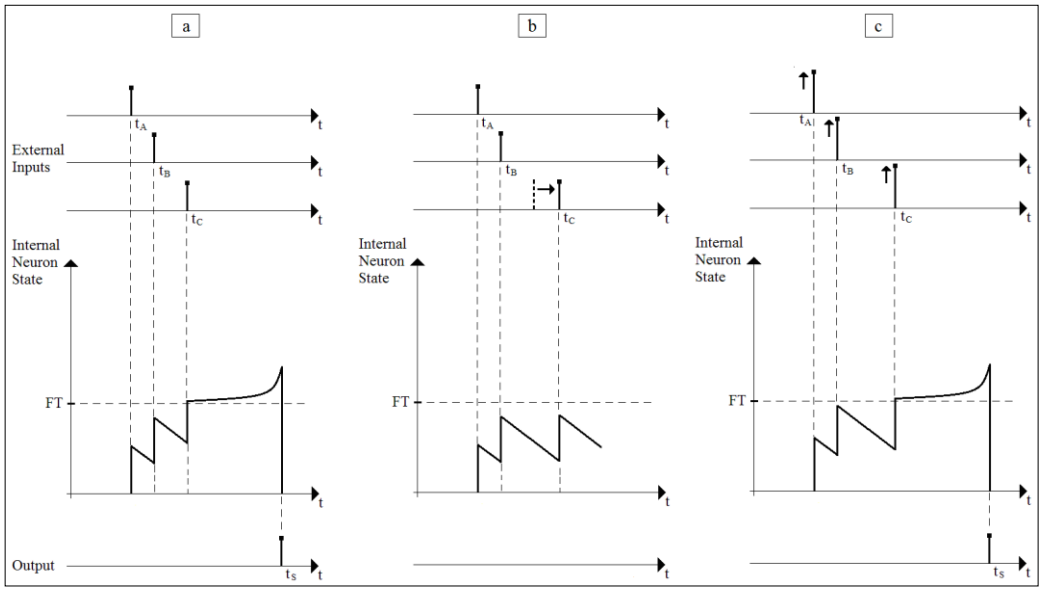


Fig.29: Paragone tra situazioni comprendenti differenti ampiezze di ingresso e tempi di arrivo, in relazione alla figura precedente.
a) Gli ingressi sono abbastanza sincronizzati e lo spike in uscita viene generato
b) Il sincronismo non è sufficiente per la generazione di spike in uscita.
c) L'insufficiente sincronismo è compensato da un determinato incremento dell'ampiezza.

Affinchè il neurone target generi uno spike, deve risultare soddisfatta la seguente relazione:

$$P_r(P_a + P_b + P_c) - K_d(\tau_1 + \tau_2) > S_0 \qquad (18)$$

in cui P_r è il *peso presinaptico*, P_a, P_b, P_c sono i *pesi postsinaptici* riferiti al *neurone target*, K_d è la *costante di decadimento sottosoglia*, S_0 è la soglia normalizzata e τ_1, τ_2 sono gli intervalli tra spike (*interspike intervals*).

La tolleranza degli impulsi di ingresso, è strettamente legata al decadimento sottosoglia e ai valori dei pesi presinaptici, considerati come parametri di controllo esterni. Sulla base di una scelta appropriata di questi parametri, può essere prevista una certa tolleranza sul sincronismo dei segnali d'ingresso, opportunamente dimensionabile: una maggiore tolleranza potrà essere ottenuta aumentando i pesi presinaptici P_r, o diminuendo il valore della costante di decadimento K_d (figura 30). Tale comportamento è di grande utilità nella realizzazione di strutture più complesse.

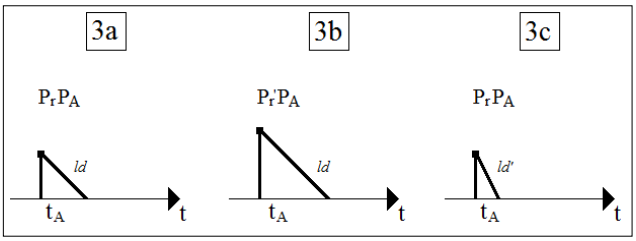


Fig.30: Vengono mostrati i diversi comportamenti del neurone target in funzione di differenti scelte di *Kd* e P_r, in zona passiva.
a) Caso di riferimento.
b) Aumentando il valore di P_r .
c) Aumentando il valore di *Kd.*

Per il target neuron può essere definito un altro utile parametro, il *livello di attività*, o *working mode activity level* (*WMAL*). Questo parametro rappresenta il numero di spike in ingresso, sincronizzati e di uguale ampiezza, necessari per mandare il *TN* sopra il livello di soglia (figura 31).

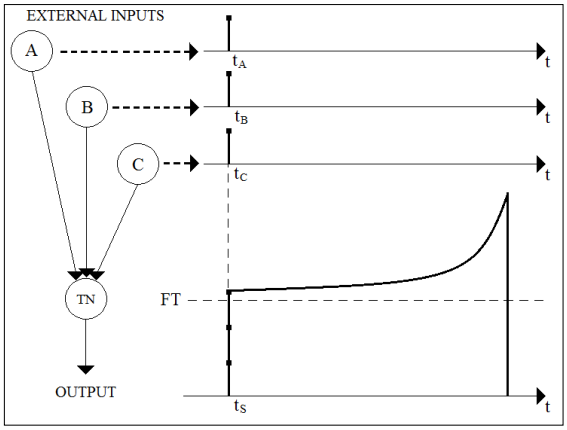


Fig.31: Caso di WMAL pari a 3.Tre impulsi uguali e sincronizzati sono necessari per mandare il neurone target in spike. Un minor numero di impulsi, oppure un sincronismo meno stretto tra gli ingressi può fare in modo che la generazione di spike non avvenga.

In fase di sintesi, ipotizzando pesi afferenti di uguale valore, il *WMAL* desiderato si ottiene dal confronto tra il prodotto tra i pesi $P_i = (P_r P_{wi})$ afferenti al neurone in questione (considerando solo burnings atti alla stessa sommazione) e il valore di soglia (indicato come S_0):

per ottenere $WMAL = 2$, bisogna imporre

$$\frac{S_0}{2} < P < S_0 \tag{19}$$

per ottenere $WMAL = 3$, bisogna imporre

$$\frac{S_0}{3} < P < \frac{S_0}{2} \tag{20}$$

…

per ottenere $WMAL = n$, bisogna imporre

$$\frac{S_0}{n} < P < \frac{S_0}{n-1} \tag{21}$$

4.1.1 – Indeterminazione istante-ampiezza

Logicamente, per i fenomeni di decadimento sottosoglia e incremento soprasoglia dello stato, e siccome il sistema è tempo-continuo, si potrà raggiungere un determinato stato, in un determinato tempo, in infinite maniere da parte del neurone: lo stato del neurone al tempo t_0 non porta informazioni precise sull'attività precedente. Progettando delle strutture atte al riconoscimento di sequenze con ampiezza non costante, oppure con possibilità di errore temporale sugli spike in ingresso, si è soggetti al fenomeno dell'indeterminazione "istante-ampiezza". Tale fenomeno introduce un grado di libertà, "dannoso" per i calcoli, sia soprasoglia che sottosoglia.

Per evidenziare questo aspetto, e per estrarre formule che potranno essere utili in seguito, verranno espressi ora, nei due modi di funzionamento del neurone, lo *stato successivo* in funzione dello *stato precedente*, entrambi facenti parte della stessa *evoluzione dello stato*.

4.1.1.1 – Indeterminazione sottosoglia

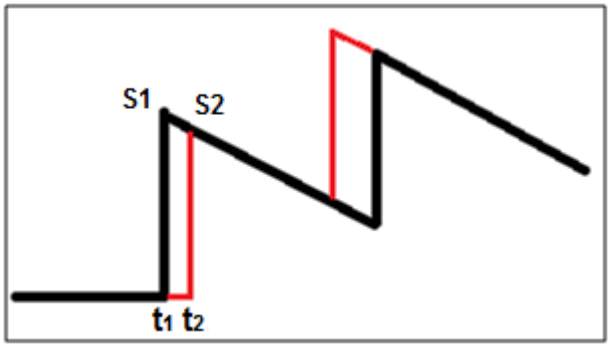


Fig.32: Fenomeno dell'indeterminazione sottosoglia: supponendo che al tempo t2 lo stato interno del neurone sia S2, c'è indeterminazione sull'attività che ha generato tale situazione. Si traccia, per evidenziare tale indeterminazione, la "curva aumentata sottosoglia"(evoluzione dello stato).

Quindi, detta:

$$S_1 = S(t_1) \tag{22}$$

e

$$S_2 = S(t_2) \tag{23}$$

allora

$$S_2 = S_1 - K_d\,(t_2 - t_1) \tag{24}$$

4.1.1.2 – Indeterminazione soprasoglia

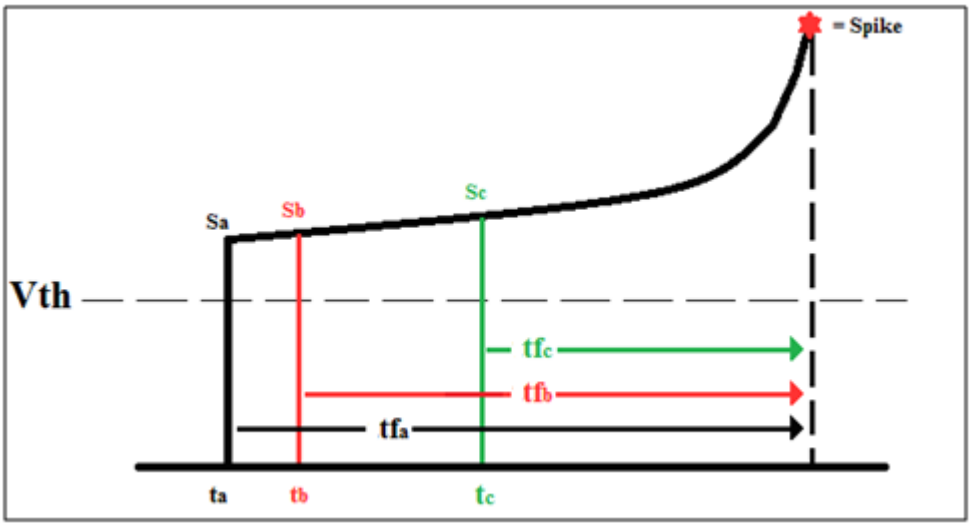


Fig.33: Fenomeno dell'indeterminazione soprasoglia: supponendo che al tempo tc lo stato interno del neurone sia Sc, c'è indeterminazione sull'attività che ha generato tale situazione. Nel grafico viene rappresentata la "curva aumentata soprasoglia" (evoluzione dello stato).

E quindi:

detta

$$S_a = S(t_a) \tag{25}$$

e

$$S_b = S(t_b) \tag{26}$$

allora avrò

$$t_{f_A} = \frac{1}{S_a - 1} \tag{27}$$

$$t_{f_B} = \frac{1}{S_b - 1} \tag{28}$$

dove

$$t_{f_B} = t_{f_A} - (t_b - t_a) = t_{f_A} - t_b + t_a = \frac{1}{S_a - 1} - t_b + t_a \tag{29}$$

e quindi:

$$S_b = 1 + \frac{1}{t_{f_B}} = 1 + \frac{1}{\frac{1}{S_a - 1} - t_b + t_a} \tag{30}$$

In seguito non sempre si incontrerà la necessità di specificare sia tutte le ampiezze degli impulsi, che tutti gli intervalli temporali consecutivi tra di essi, per individuare univocamente una sequenza. Potrà quindi essere utile far riferimento alle relazioni ricavate in questo paragrafo.

4.2 – Strutture elementari

In questa sezione sono presentate applicazioni elementari del modello neuronale realizzato, ottenibili sfruttando i fenomeni precedentemente descritti. Lo scopo è quello di introdurre delle strutture in grado di generare o rivelare determinate sequenze di spike. Visto che tale classe di reti è in grado di generare sequenze costituite da *timing* a carattere analogico, il tempo verrà considerato come una quantità continua in tutte le applicazioni. Verranno inoltre descritti i principi di realizzazione e le tecniche di dimensionamento per la sintesi di tali strutture funzionali.

Per convenzione, si indicherà con *P(a, b)* il prodotto tra i pesi presinaptico e postsinaptico relativo alla sinapsi *a - b.*

4.2.1 – Generatori di sequenze

In questa sezione vengono definite strutture neurali chiamate *generatori di sequenze*. Tali strutture permettono la propagazione di uno spike all'interno di un percorso composto da neuroni, generando timings caratteristici.

4.2.1.1 – *Open neural chain*

La *open neural chain* consiste in gruppi funzionali di *m* neuroni (N_1, $N_2, \dots, N_m$), connessi in sequenza, in modo che la sinapsi tra il neurone N_k al neurone N_{k+1} sia sempre presente, con $k = 1, 2, 3, \dots, m-1$ (figura 34).

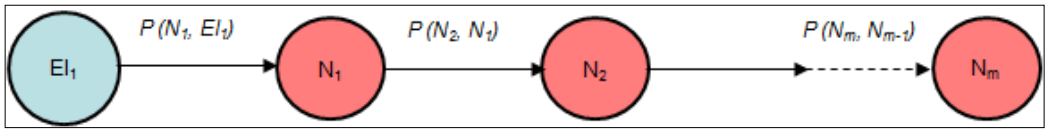


Fig.34: *Open neural chain*.

Nella figura, EI_1 rappresenta l'input esterno, N_k il neurone k e $P(N_k, N_{k-1})$ il prodotto tra i pesi presinaptici e postsinaptici, per le sinapsi che vanno dal neurone N_{k-1} al neurone N_k.

Se tutte le quantità $P(N_k, N_{k-1})$ risultano maggiori di un valore di soglia

$$S_0 = 1 + d \tag{31}$$

ogni neurone diverrà attivo in presenza di un singolo spike di ingresso dal precedente neurone nella catena.

Questo tipo di propagazione, in virtù di quanto detto nei precedenti paragrafi, viene definita di tipo *WMAL1:* affinchè il firing proceda

lungo la catena, per ogni neurone la generazione di un singolo spike deve essere sufficiente a rendere attivo il successivo, se presente.
Infatti, solo se tutti i neuroni *1*, *2*, .. *m*, sono di tipo *WMAL1*, la sequenza di firing può procedere da 1 a m.
Vedremo che, operando qualche piccolo accorgimento a questa struttura, si può fare in modo che la sequenza di firing si ripeta.

4.2.1.2 – Closed neural chain

Aggiungendo una ulteriore connessione sinaptica dal neurone N_m ad uno dei neuroni della catena, la sequenza di firing può ripetersi indefinitamente, realizzando una *closed neural chain* (figura 35).

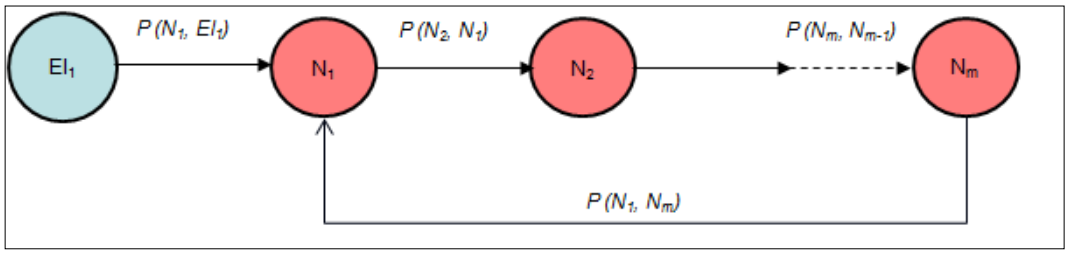


Fig.35: Closed neural chain.

Sia in questa struttura che nella precedente, l'informazione caratterizzante la catena di neuroni consiste nella sequenza dei timings che essa genera, e risulta "codificata" nei pesi delle connessioni di cui è composta.
Infatti, al variare delle quantità $P(N_k, N_{k-1})$ e S_0 scelte per il modello, verranno generati differenti *latency times*: i timings possono quindi essere modulati, per ogni *k*, con una opportuna scelta di $P(N_k, N_{k-1})$.

In questo modo può essere ottenuta una desiderata sequenza di spiking times, che sarà limitata per *open neural chain*, periodica per *closed neural chain*.

4.2.1.3 – Extended neural chain element

Le neural chain appena viste, possono essere modificate in molti modi. Una semplice idea consiste nel sostituire un neurone N_k con un set di altri neuroni N_{kh}, con $h = 1, 2, \ldots m_k$, ($m_k > 1$) (figura 36), in modo da conferire alla sequenza particolari proprietà utili in varie applicazioni.

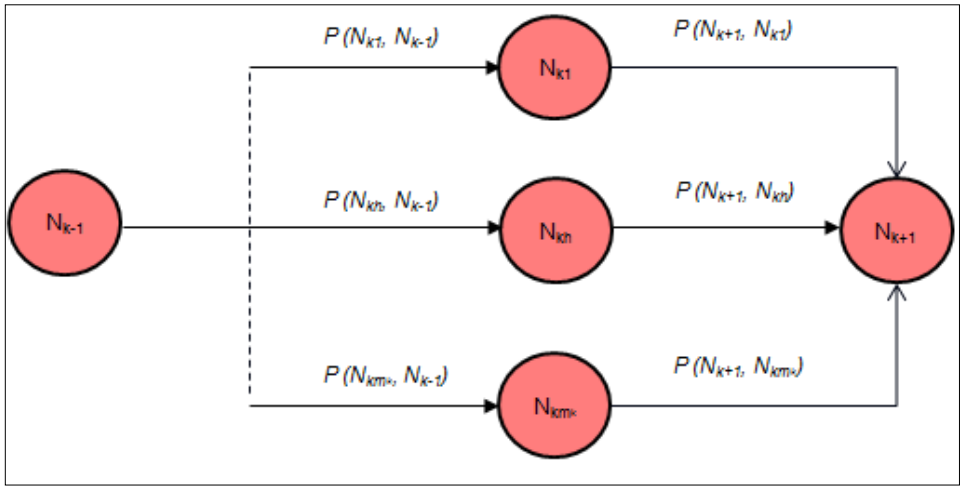


Fig.36: Extended neural chain element.

4.3 – Spike-timing sequence detectors

Come gia ampiamente spiegato nel primo capitolo, alcuni studi sperimentali di decodifica di segnali neurali del cervello affermano

che i neuroni trasmettono informazioni atteraverso rate-coding, ma recenti studi sulla computazione neurale indicano che il codice temporale è uno schema biologicamente plausibile utilizzato probabilmente in parecchie zone del sistema nervoso. Ma con gli effetti a disposizione dal modello realizzato, come possono essere rilevati tali intervalli?

In questa sezione vengono presentate delle strutture capaci di individuare determinati intervalli tra gli impulsi in arrivo; la segnalazione di tale rivelazione da parte delle strutture viene indicata dall'attivazione di un opportuno neurone target.

Il funzionamento delle strutture di seguito presentate, verrà chiarito tramite un *diagramma temporale*, e la relazione tra l'accensione del neurone target e l'arrivo degli spike sull'asse degli ingressi sarà volta per volta mostrata tramite il *dominio di attivazione* (*singolo* o *multiplo*).

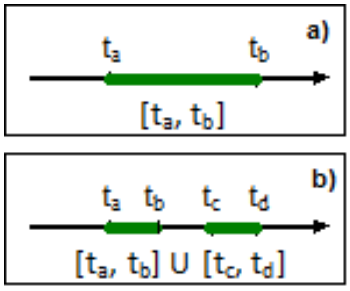


Fig.37: Domini di attivazione di tipo *singolo* (a) e *multiplo* (b)

Per tutte le strutture illustrate si considerano ingressi di uguale ampiezza ed unitari (a meno che non espressamente indicato), in modo da poter fare un'analisi del comportamento delle strutture a parità di condizioni iniziali.

Esistono, per tali strutture, varie possibilità realizzative, ognuna con le sue peculiarità; risulta infatti difficile la loro classificazione in quanto esse possono tra loro differire per tipologia di *dominio di attivazione*, numero di linee di ingresso contemplate, numero e tipo di neuroni coinvolti, o numero di effetti considerati.
Nel corso di questa trattazione si cercherà di seguire un ordine di presentazione delle strutture basato sulla loro complessità di funzionamento, cosicchè il lettore si possa rendere conto delle migliorie conseguibili da un consapevole uso degli effetti implementati, e delle innumerevoli possibilità ottenibili.

4.3.1 – Direct *spike-timing sequence detector*

Questa è la più semplice struttura con cui è possibile rilevare particolari intervalli nelle sequenze di ingresso, con l'opportunità di intervenire, eventualmente, in maniera diretta sulla tolleranza voluta.
Le proprietà della struttura dipendono dalla scelta dei *pesi postsinaptici* delle sinapsi direttamente afferenti al target e dall'effetto di *decadimento sottosoglia*. Tale struttura è in grado di funzionare accettando gli impulsi su linee di ingresso differenti (in generale *n*).

4.3.1.1 – Simple direct *spike-timing sequence detector*

L'idea di base è basata sulla rete mostrata nella seguente figura: essa consiste in due rami connessi agli input esterni (EI_1 , EI_2) ed afferenti ad un neurone target (*TN*). Tale struttura è in grado di funzionare accettando gli impulsi su 2 linee di ingresso differenti (sequenze di impulsi di cardinalità 2).

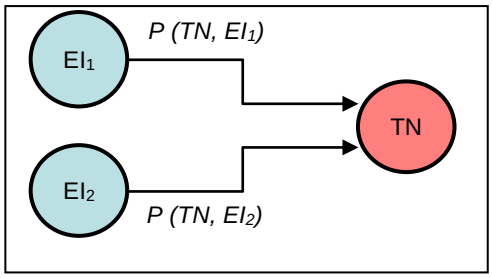


Fig.38: Direct Spike Timing Sequence Detector a 2 ingressi. La rivelazione è positiva quando il neurone TN produce uno spike; altrimenti è negativa.

Nella figura, $P(TN, EI_1)$ è il prodotto tra i pesi presinaptico e postsinaptico relativi alla sinapsi EI_1 - *TN*, mentre $P(TN,EI_2)$ è il prodotto riferito alla sinapsi EI_2 - *TN*. Con una opportuna scelta di tali valori, il sistema può lavorare con un *working mode level* pari a 2. Questo significa che *TN* può diventare attivo in presenza di due spikes dagli input EI_1 ed EI_2. Per esempio, se $P(TN, EI_1) = P(TN, EI_2) = 0.8$, e gli spikes da EI_1 and EI_2 arrivano in tempi molto vicini, *TN* diviene attivo con stato pari a

$$S(TN) = 1.6 \; (> S_0) \tag{32}$$

tale che in uscita venga prodotto uno spike, dopo un opportuno intervallo di *latency*. In questo caso, lo spike di uscita generato da *TN* rivela che la struttura ha ricevuto in ingresso una sequenza costituita da due spike distanziati da tale intervallo. Comunque, se i tempi di

firing t_1 e t_2 relativi a EI_1 e EI_2 sono molto differenti, (per esempio $t_2 > t_1$), solo il valore di $S\,(TN) = 0.8$ è presente al tempo t_1. Prendendo in considerazione il fenomeno di *decadimento sottosoglia*, lo stato interno del neurone *TN* al tempo t_2 decadrà a:

$$S\,(TN) = 0.8 - (t_2 - t_1)\,K_d \qquad (33)$$

Allora, il nuovo spike in arrivo da EI_2 non ce la farà a rendere attivo il neurone *TN*; in questo caso, la rivelazione sarà negativa. Questo è vero se i valori di $P(TN, EI_1)$ e $P(TN, EI_2)$, l'intervallo $(t_2 - t_1)$ e la costante di decadimento lineare K_d sono opportunamente scelte. Ad ogni modo, la rivelazione è ancora positiva se $(t_2 - t_1)$ è abbastanza limitata.

Consideriamo per semplicità, qui e in tutte le successive trattazioni, che l'ingresso di EI_1 preceda quello EI_2 (ovvero $t_2 > t_1$), come mostrato schematicamente in figura.

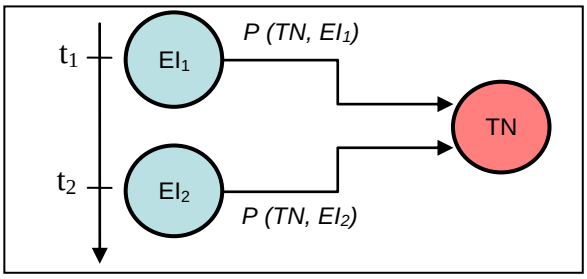
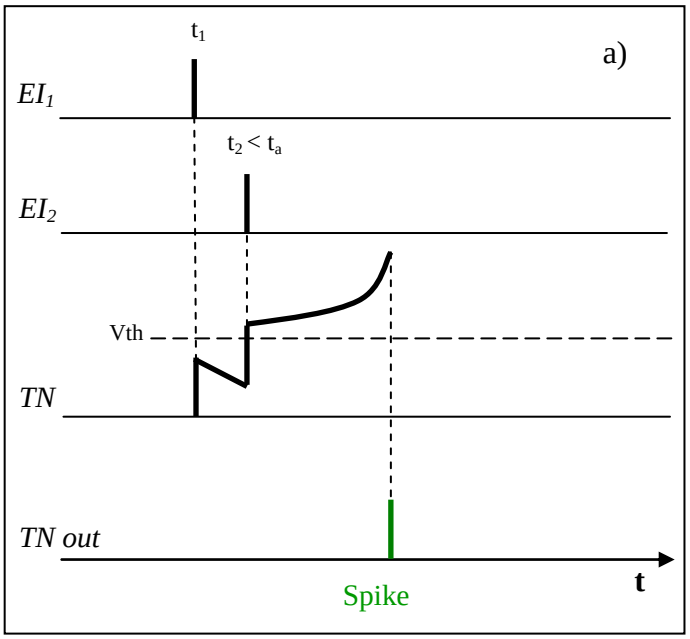


Fig.39: Direct Spike Timing Sequence Detector a 2 ingressi; gli external input vengono investiti in maniera sequenziale dagli impulsi della sequenza proposta.

Si tracciano di seguito il *diagramma temporale* (figura 40 a, b) e il *dominio di attivazione* (figura 41), supponendo i pesi presinaptici di valore unitario, e quelli postsinaptici $P(TN, EI_1) = P(TN, EI_2) = 0.8$



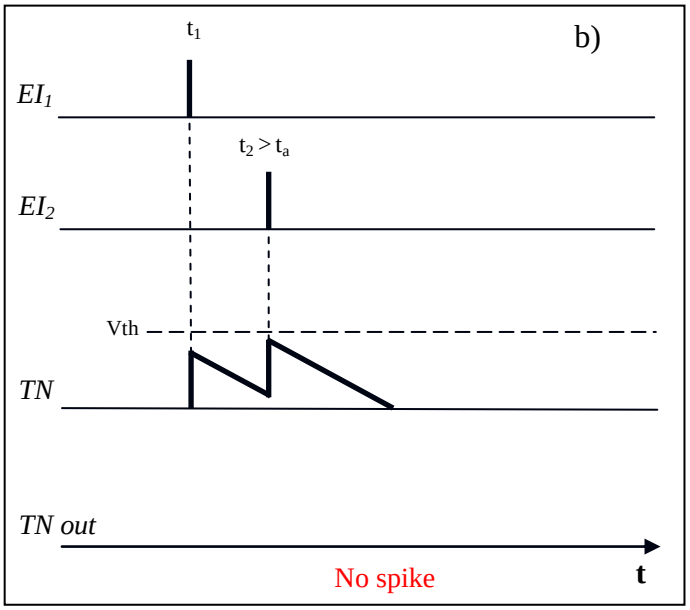


Fig.40: Diagramma temporale del *simple direct spike-timing sequence detector*. Come si vede dalla tavola a), per un intervallo contenuto tra il primo ed il secondo impulso di ingresso, il target riesce a superare la soglia, generando uno spike in uscita; per un intervallo maggiore, raffigurato nella tavola b), si nota come, a causa del fenomeno di *decadimento sottosoglia*, la rivelazione diviene negativa.

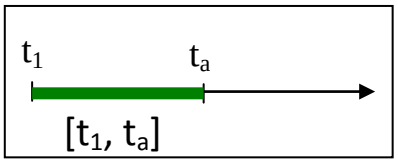


Fig.41 : *Dominio di attivazione* della struttura presentata: per intervalli compresi tra 0 ed un certo valore t_a, la rivelazione risulti positiva, negativa altrimenti. Il dominio risulta *continuo*.

Avendo analizzato in maniera approfondita la struttura, si eliminano alcuni gradi di libertà per rendere più semplice la progettazione. Sono mostrate di seguito le due relazioni per effettuare il dimensionamento della rete. La prima ci assicura che il target superi la soglia:

$$P_r P_T - (t_2 - t_1) K_d > 1 + K_{th} \tag{34}$$

e, la seconda, che esso lavori a *WMAL* = 2:

$$\frac{1}{2} \frac{1+K_{th}}{P_r} < P_T < \frac{1+K_{th}}{P_r} \tag{35}$$

con P_T valore dei pesi afferenti al target (uguale per tutti) e P_r valore dei pesi presinaptici (parametro globale).

4.3.1.2 – Multi branch direct *spike-timing sequence detector*

Eliminando l'ipotesi di ingressi di uguale ampiezza, può essere immaginata una nuova configurazione in grado di poter individuare una sequenza di numerosità maggiore di 2; a questo proposito può essere introdotta la nuova struttura illustrata in figura 42

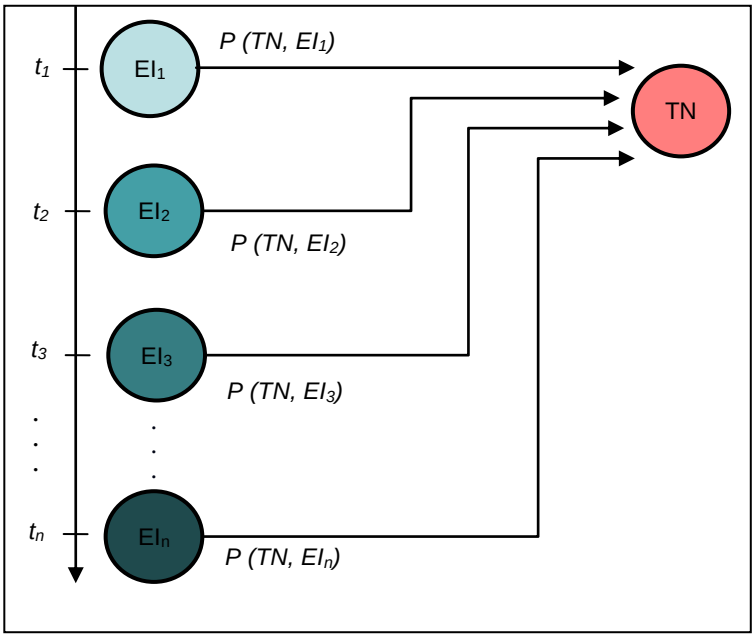
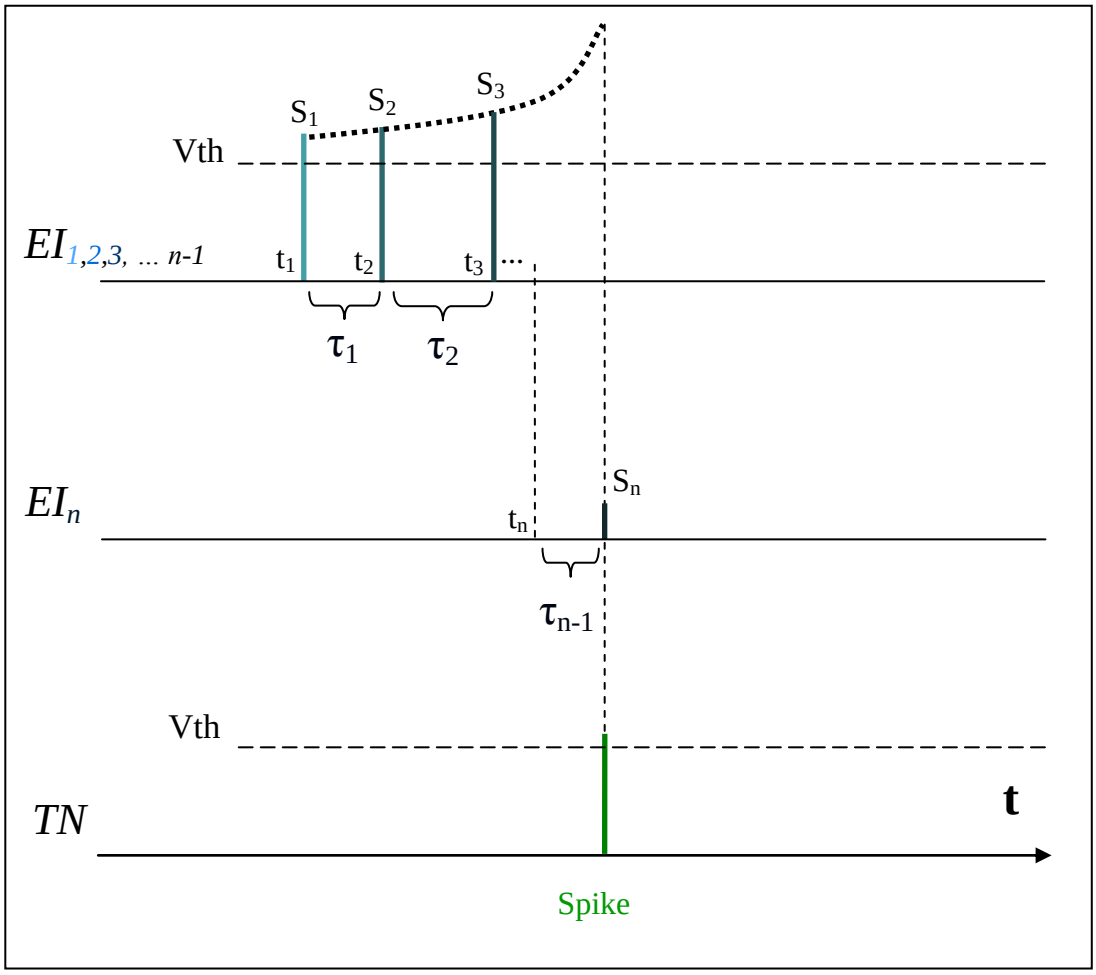


Fig.42 : *Multi branch direct* spike-timing sequence detector

Il funzionamento si basa sul fatto che gli istanti di presentazione degli ingressi debbano ricalcare la medesima curva attiva (si veda par. 4.1.1) generata dal primo di essi, EI_1, agendo in parallelo e concorrendo per una sommazione sincrona sul target.

Per semplicità di progettazione, elimineremo un grado di libertà, attribuendo valore unitario al peso $P\ (TN, EI_n)$.

Ipotizzando che la sequenza che si vuole rivelare sia caratterizzata da una successione di intervalli $\tau_1,\ \tau_2,\ \ldots,\ \tau_{n-1}$, lo scopo che si vuole ottenere è la possibilità di rivelazione con una certa tolleranza, espressa nella trattazione dal parametro TOL .



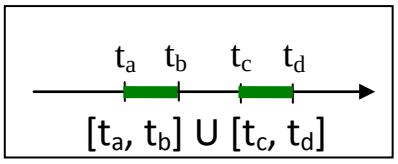


Fig.43 : *Diagramma temporale* e *dominio di attivazione* della struttura presentata: si nota come essa esibisca un dominio *multiplo.*

In fase di progettazione, in virtù del suo principio di funzionamento, tale struttura dovrà essere tarata su ingressi che presentino ampiezze crescenti da linea *1* a linea *n-1*; la linea *n*, a differenza delle altre, concorrendo direttamente al target, avrà un'ampiezza attesa di valore pari a:

$$P\,(TN, EI_n) \;=\; 1 \qquad (36)$$

I vincoli riguardano essenzialmente la natura degli ingressi e la tolleranza massima utilizzabile per rendere la struttura più o meno selettiva; definendo la grandezza *MaxTol* come tolleranza massima accettabile sull'arrivo degli impulsi in ingresso, affinchè sia prodotto uno spike dal neurone target, si ha:

$$-k_d = \frac{0-\left[\frac{1+k_{th}}{P_r(n-1)} - \frac{1+k_{th}}{nP_r}\right]}{MaxTol-0} \tag{37}$$

e quindi:

$$MaxTol = \left|\frac{\left[\frac{1+k_{th}}{P_r(n-1)} - \frac{1+k_{th}}{P_r n}\right]}{k_d}\right| \tag{38}$$

Il calcolo deriva dalla seguente figura:

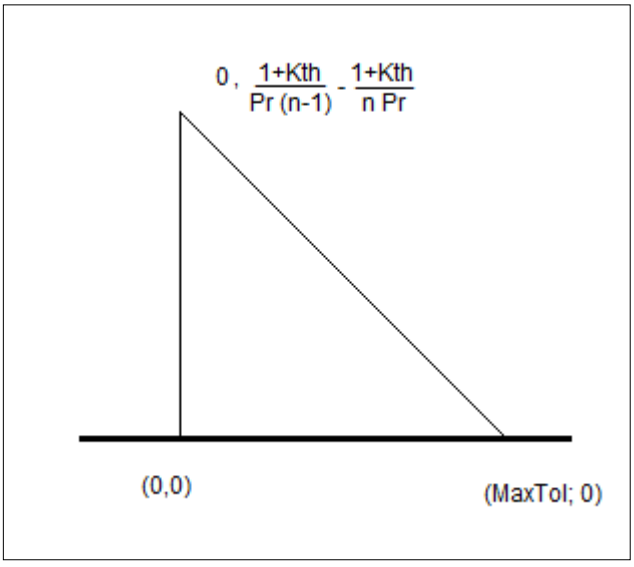


Fig.44 : Calcolo del valore *MaxTol* in relazione alla costante di decadimento sul neurone target

Per quanto riguarda la natura degli ingressi, i vincoli da rispettare sono i seguenti:

$$Iext_{n-1} > Iext_{n-2} > \cdots > Iext_1 \tag{39}$$

$$0 < TOL < MaxTol \tag{40}$$

Si procede ora al calcolo dei diversi parametri per definire la struttura:

- 1°STEP

  Calcolo del τ_{TOT} $(= \tau_1 + \tau_2 + \cdots + \tau_{n-1})$

  Il vincolo è il seguente:

$$\tau_{TOT} < \frac{1}{k_{th}} - TOL \tag{41}$$

- 2°STEP

  Calcolo delle intensità $Iext$ tali che sia verificata la condizione di sincronismo al target.

  Si ha:

$$S_1 = 1 + \frac{1}{\tau_{TOT}} \tag{42}$$

$$S_2 = 1 + \frac{1}{\frac{1}{S_1 - 1} - \tau_1} \tag{43}$$

$$S_3 = 1 + \frac{1}{\frac{1}{S_2 - 1} - \tau_2} \tag{44}$$

.
.
.
.

$$S_{n-1} = 1 + \frac{1}{\frac{1}{S_{n-2}-1} - \tau_{n-2}} \tag{45}$$

  Naturalmente, $S_n = P_T$, e verrà ricavato successivamente

  Da notare che tutti gli *S* calcolati ricalcano la curva immaginaria generata dal primo impulso, trovata, quindi, a partire dal τ_{TOT}.

- 3°STEP

  Calcolo di $P_T = S_N$. A tale scopo, ci si basa sulla seguente immagine:

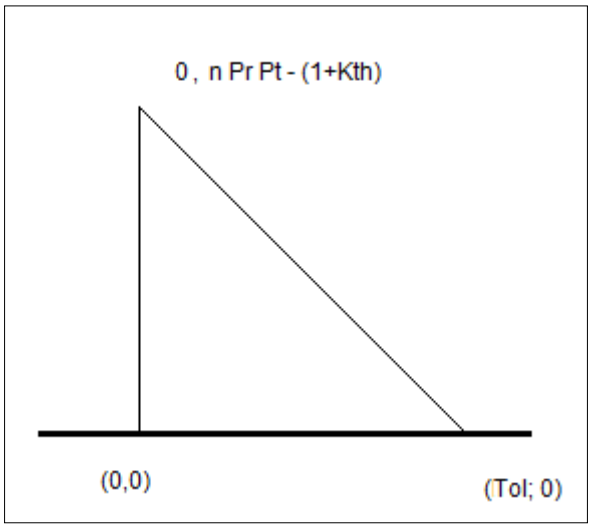


Fig.45 : Calcolo del valore *Pt* in relazione al decadimento sottosoglia dello stato del neurone target.

Dalla figura si ha:

$$-k_d = \frac{[nP_r P_T - (1+k_{th})] - 0}{TOL - 0} \tag{46}$$

e quindi:

$$P_T = \left| \frac{-Kd\ TOL + (1+Kth)}{(n\ Pr\ )} \right| \tag{47}$$

Dove per TOL si intende:

$$TOL = |max(scostamenti\ in\ eccesso)| + |max\ (scostamenti\ in\ difetto)| \tag{48}$$

Per una maggiore comprensione, vengono riportate nell'*Appendice* alcune prove di dimensionamento della struttura.

Se da un lato è stato un bene svincolare le relazioni dall'ipotesi di ingressi unitari, per rendere la struttura in grado di poter individuare

domini *multipli*, dall'altro la struttura è stata resa suscettibile di indeterminazione istante-ampiezza (si veda par. 4.1.1), in quanto non è in grado di distinguere tra l'arrivo di un impulso ad ampiezza attesa nell'istante atteso, ed uno con ampiezza diversa ma il cui istante di arrivo ricalchi la stessa *evoluzione dello stato*.

4.3.2 – Delayed *spike-timing sequence detector*

Questa è un'altra struttura con cui è possibile rilevare particolari intervalli nelle sequenze di ingresso, ed intervenire in maniera diretta sulla tolleranza voluta.

La struttura è caratterizzata dalla comparsa del delay neuron, neurone di tipo *eccitatore* in grado di creare un ritardo di trasmissione su una o piu linee e capace di generare fenomeni che possono essere sfruttati per ottenere particolari effetti. Le proprietà della struttura dipendono dall'effetto di *decadimento sottosoglia*, dalla scelta dei *pesi postsinaptici* delle sinapsi direttamente afferenti al target e di quelle afferenti ai neuroni di delay. Si vedrà piu avanti che anche tale struttura è in grado di funzionare accettando gli impulsi su linee di ingresso differenti (in generale *n*).

4.3.2.1 – Simple delayed *spike-timing sequence detector*

La precedente trattazione sulle simple *direct spike-timing sequence detector* può essere facilmente estesa per definire domini di attivazione più complessi . Se la struttura viene modificata come in figura 46, dove è presente un *delay neuron* (*DN*), continuando ad utilizzare l'ipotesi di ingressi di uguale ampiezza per una facilitata progettazione, il dominio di attivazione può essere traslato attorno al valore $t_1 + Dt$, ove *Dt* è il tempo di delay relativo alla *latency* ottenuta in *DN*.

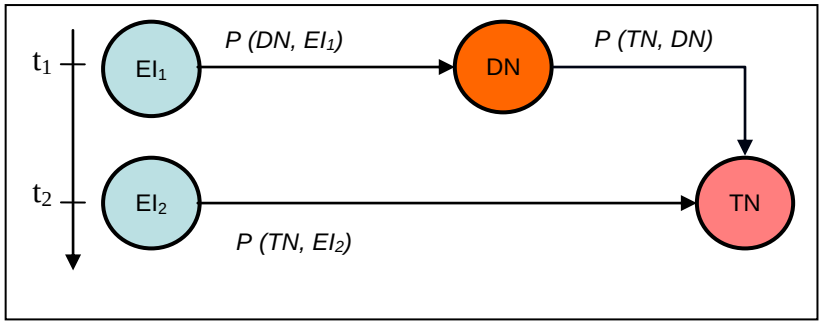
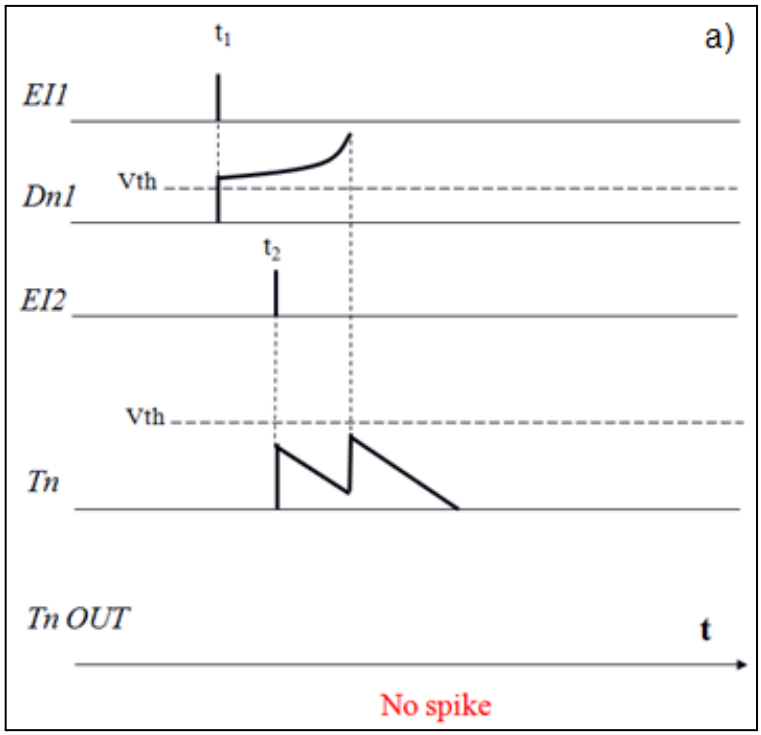
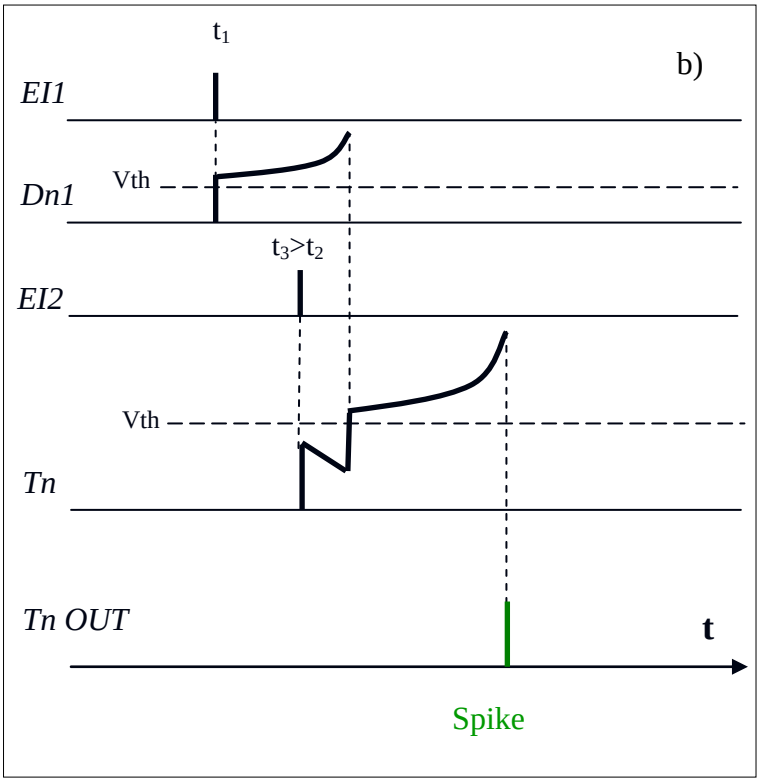


Fig.46: Simple delayed spike-timing sequence Detector: struttura.

Il valore di *Dt* può essere ottenuto da opportuni valori dei *pesi*. Sicuramente tale neurone dovrà lavorare a *WMAL* pari a *1*.

Nella seguente figura viene riportato il principio di funzionamento della struttura, e nella figura seguente il dominio di attivazione ottenibile.


a)
t1
EI1
Dn1
Vth
t2
EI2
Vth
Tn
Tn OUT
t
No spike


b)
t1
EI1
Dn1
Vth
t3>t2
EI2
Vth
Tn
Tn OUT
t
Spike

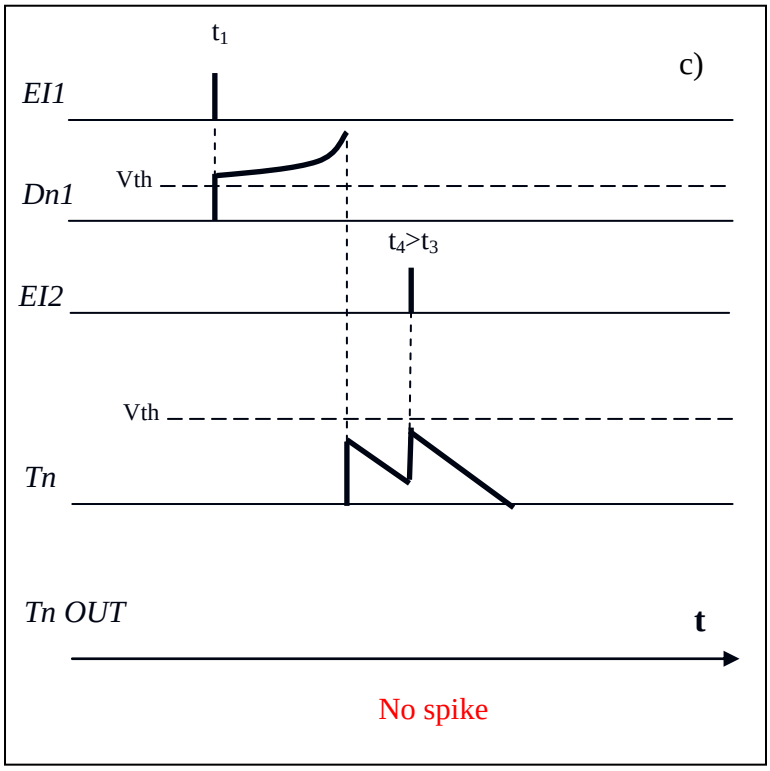


Fig.47: Si nota come, al variare dell'intervallo tra gli ingressi presentati, la struttura è in grado di esibire uno spike in uscita solo per valori compresi in un certo range, attorno ad un determinato valore temporale diverso da t_1.

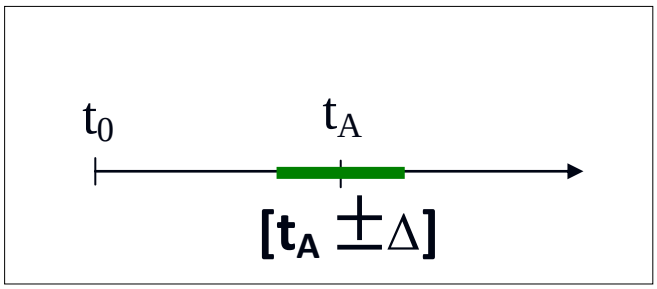


Fig.48: *Dominio di attivazione* per la architettura presentata.

I requisiti necessari per il corretto funzionamento della struttura sono i seguenti:

- Per quel che concerne *Dn*: *WMAL* = 1
- Per quel che concerne *Tn*: *WMAL* = 2

quindi:

- per assicurare l'attivazione dei neuroni dello strato di ingresso, deve risultare:

$$EI1 \,\&\, EI2 \;>\; 1 + Kth \tag{49}$$

$$P(DN, EI_1) >\; 1 + K_{th} \tag{50}$$

- Per assicurare l'attivazione del neurone DN deve risultare:

$$P_r\; P(DN, K_{th}) >\; 1 + K_{th} \tag{51}$$

e cioè

$$P(DN, EI_1) > \frac{1+K_{th}}{P_r} \tag{52}$$

- Per assicurare lo spike in uscita, nella favorevole condizione di sincronismo sul target, dev'essere:

$$Pr\; P(TN, DN) +\; P(TN, EI_2) \;>\; 1 +\; K_{th} \tag{53}$$

$$P(TN, DN) +\; P(TN, EI_2) > \frac{1+\,K_{th}}{P_r} \tag{54}$$

Da cio si deduce che la condizione per cui si avrà spike al target è la seguente (consultare l'appendice per i calcoli intermedi):

$$\Pr[P(TN, EI_2) + P(TN, DN)] - K_d \left|\frac{1}{(Iext_2 - 1)} - \frac{1}{Iext_1 - 1} - \frac{1}{[Pr\; P(DN, EI_1) - 1]} + \tau_{in}\right| > (1 + K_{th}) \tag{55}$$

Un'osservazione sulla ultima relazione scritta riguarda il fatto che il presente sviluppo ha l'obiettivo di poter individuare delle relazioni che possano permettetere di progettare strutture di questo tipo per specifici scopi; da tale disequazione, però, si nota la quantità esagerata gradi di libertà, incompatibile con la progettazione della struttura in modo univoco.

Affinché essa supporti la condizione di sincronismo dei 2 contributi sul target, dovrà essere verificata l'uguaglianza:

$$t_f(EI_1) + t_f(DN) = \tau_{in} + \tau_{out} + t_f(EI_2) \quad (56)$$

con $\tau_{out} = 0$. Allora si riscrive:

$$t_f(EI_1) + t_f(EI_2) = \tau_{in} + t_f(EI_2) \quad (57)$$

Va da sé che per *spike timing sequences* troppo distanziate, tale struttura sarebbe inutile. In quel caso servirà aggiungere, in cascata ai neuroni del ramo superiore, altri neuroni *A3*, *A4*, … , affinchè la relazione:

$$t_f(EI_1) + t_f(DN_1) + t_f(DN_2) + \cdots + t_f(DN_n) = \tau_{in} + t_f(EI_2) \quad (58)$$

venga soddisfatta.

Per riconoscere sequenze di numerosità maggiore (n), basterà aumentare il numero dei rami di tipo *A*, facendo in modo che i contributi arrivino simultaneamente al neurone target, il quale dovrà essere del tipo *WMAL m.*

Per quel che riguarda, invece, la sintesi della struttura, i parametri di progetto saranno:

- τin = intervallo relativo alla sequenza da riconoscere.
- *Tolerance* (*TOL*) = errore ammissibile sul tempo di arrivo del secondo impulso (in modo da mandare in spike il target neuron finchè la sequenza presenta un errore: $\tau_{in} \pm TOL$)
- $Iext_1$, $Iext_2$: valori attesi delle ampiezze degli ingressi.
- K_d , K_{th} , P_r

La progettazione inizia col calcolo immediato della *max Tolerance* (*MaxTol)* supportata a partire dal K_d considerato; dalla relazione (73) riportata più avanti, si ottiene:

$$K_d = \frac{(2P_r P_T - 1 - K_{th})}{Tol} \tag{59}$$

ponendo P_T all'estremo superiore (si veda (67)):

$$K_d = \frac{\left\{2P_r \frac{1+K_{th}}{P_r} - 1 - K_{th}\right\}}{MaxTol} \tag{60}$$

e allora

$$MaxTol = \frac{[2\,(1+K_{th}) - 1 - K_{th}]}{K_d} = \frac{(1+K_{th})}{K_d} \tag{61}$$

A questo punto i requisiti necessari per il funzionamento dettano:

$$Iext_1 \;\&\; Iext_2 > 1+K_{th} \tag{62}$$

$$P(DN, EI_1) > 1+K_{th} \tag{63}$$

Invece per l'eliminazione dei gradi della struttura si dovrà agire in questo modo:

- dosare uniformemente il *working mode level* del neurone Target rispetto al numero di afferenze (cioè pesi postsinaptici uguali afferenti a *TN*):

$$Pr\,P(TN, EI_2) = Pr\,P(TN, DN) = P_r\,P_T \tag{64}$$

Per assicurare lo spike in uscita, dalla (3) si ha:

$$2P_r\,P_T > (1 + Kth) \tag{65}$$

Ma per assicurare che il target non lavori in working mode level 1 dev'essere:

$$P_r\, P_T \;<\; (1 + K_{th}) \tag{66}$$

- mettendo insieme le 2 ultime condizioni, per uno *spike timing sequence detector* a 2 ingressi si ottiene:

$$1/2\ \ [(1+K_{th})/P_r] < P_T < [(1+K_{th})/P_r] \tag{67}$$

In questo modo, in condizioni di sincronismo sul target, sicuramente si avrà spike in uscita e sul Target si lavorerà in *WMAL2*.

Più P_T tende a $[(1+K_{th})/P_r]$, più il riconoscitore sarà *tollerante*, riguardo la precisione temporale del 2° impulso in ingresso.

Più P_T tende a $[(1+K_{th})/2P_r]$, meno il riconoscitore sarà *tollerante*, riguardo la precisione temporale del 2° impulso in ingresso.

4.3.2.2 – Telescopic delayed *spike-timing sequence detector*

Qualora il delta tra i due ingressi sia maggiore dal massimo intervallo supportabile dalla struttura ad un solo neurone di delay, il ramo *delayed* può essere "allungato", aggiungendo un neurone in modo che la struttura possa essere utilizzata per la rivelazione di intervalli piu lunghi. Per semplicità di calcolo, e per eliminare fuorvianti gradi di libertà, nell progettazione tali strutture saranno "forzate" ad avere i pesi dei neuroni *delayed aggiunti* di valore minimo (corrispondenti a $t_{f\,max}$), potendo così dimensionare tutto il ramo solo andando a calcolare il peso $P(DN_1,EI_1)$.

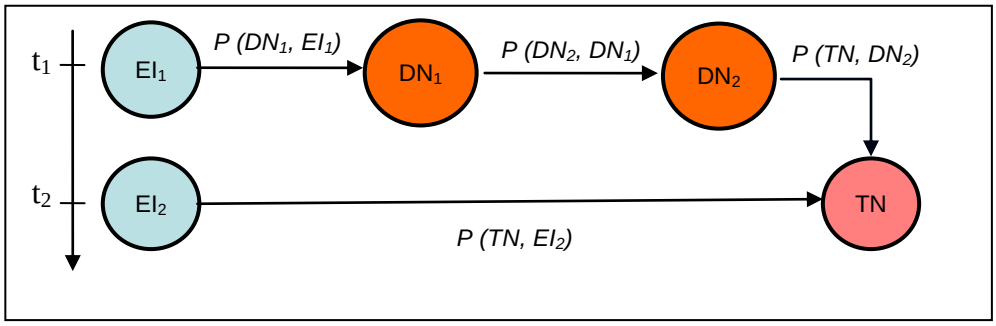


Fig.49: Struttura del *Telescopic delayed* spike-timing sequence detector

Il dominio di attivazione derivante da tale struttura sarà della stessa tipologia del delayed semplice, ma traslato in avanti grazie al nuovo neurone di ritardo inserito nel ramo.

In fase di progettazione, per il corretto dimensionamento della struttura bisogna agire come di seguito riportato:

- Calcolo preliminare del numero di neuroni necessari nel ramo superiore, in base al τin:

$$N_{delay\_line} = P_r[(\tau_{in} + t_f(EI_2) + Tol - t_f(EI_1))/MaxLat\ ] + 1 \tag{68}$$

ipotizzando che, tranne il primo del ramo, gli altri pesi siano inizializzati a tale valore:

$$P(n-1,n) = \frac{1}{[(1+K_{th})-1]P_r} \tag{69}$$

- Calcolo di $P(DN_1,EI_1)$ per ottenere le condizioni di sincronismo sul target, a partire da τ_{in} arrivando alla relazione (la trattazione completa è riportata nell'appendice):

$$P(DN_1, EI_1) = \frac{\tau_{in} - \left[\frac{1}{Iext_1 - 1}\right] + \left[\frac{1}{Iext_2 - 1}\right] - \frac{1}{K_{th}}\left(N_{delay\_line} - 1\right)}{P_r} \tag{70}$$

- Calcolo di P_T per ottenere la tolleranza desiderata, facendo uso della figura 50 (K_d è gia fissato):

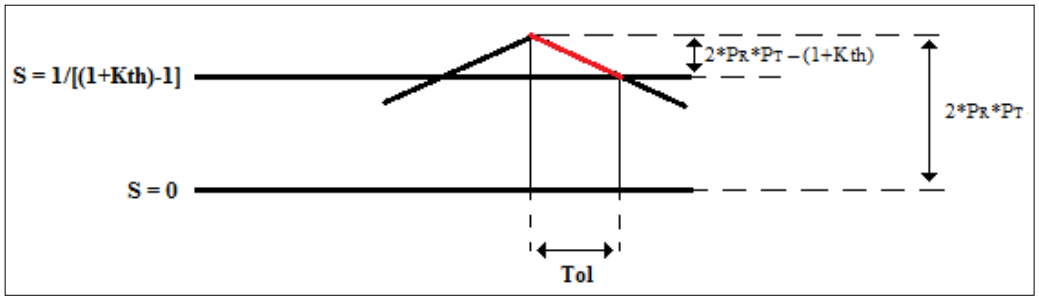


Fig.50: Sommazione contributi al Target: se i contributi arrivano in maniera sincrona, cioè in assenza di errore, il valore dello stato raggiunto dal Target è pari a *2PrPT*; se i contributi arrivano al massimo scostamento possibile, definito con la Tolerance (tramandata in uscita è la stessa), il valore dello stato raggiunto dal Target è pari a *(1+Kth).*

Ora si cerca una relazione per ricavare P_T, in funzione della tolleranza desiderata.

- Prendendo in considerazione il segmento rosso in figura 50, si scrive:

$$m = -K_d = \frac{y_2 - y_1}{x_2 - x_1} \tag{71}$$

dove con *m* si intende il coefficiente angolare.

Ancora:

$$-K_d = \frac{0 - \left[2P_r P_T - (1 + K_{th})\right]}{Tol - 0} = -\frac{(2P_r P_T - 1 - K_{th})}{Tol} \tag{72}$$

quindi:

$$K_d = \frac{(2P_r P_T - 1 - K_{th})}{\mathrm{Tol}} \tag{73}$$

e

$$P_T = \frac{K_d\, Tol + 1 + K_{th}}{2P_r} \tag{74}$$

- Adesso è possibile trovare il *time to fire* minimo del target (in condizioni di sincronismo sul target, ovvero in assenza di errore) :

$$t_{f\,min}(tgt) = \frac{1}{(\,2\,P_r P_T - 1)} \tag{75}$$

Da notare che, per applicazioni in cui si voglia univocamente riconoscere determinati ingressi, bisognerà fare in modo che alla struttura non possano arrivare impulsi ad ampiezze differenti da quelle stabilite in fase di progettazione: il fenomeno dell' "indeterminazione istante-ampiezza", ampiamente descritto nei paragrafi precedenti, potrebbe essere dannoso: si tratterebbe di un altro grado di libertà da eliminare.

4.3.2.3 – Multi branch delayed *spike-timing sequence detector*

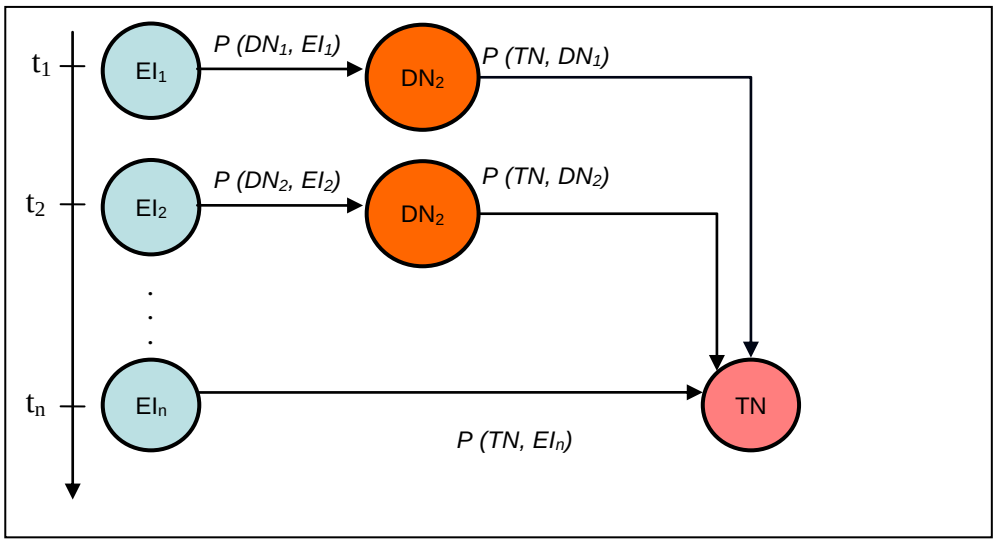


Fig.52: Struttura *Multi branch delayed* spike-timing sequence detector

Sio modifichi ora la struttura per adattarla al riconoscimento di una sequenza composta da un numero di spikes maggiore di 2 (pari a *N*).
I requisiti necessari per il funzionamento sono i seguenti:

$$Iext_1 \, , Iext_2 \, , \, \dots \, , Iext_N \, > \, (1 + K_{th}) \tag{76}$$

Si procede con l'eliminazione dei gradi di libertà della struttura:
Distribuendo uniformemente il working mode level del neurone target rispetto al numero di afferenze si ottiene:

$$P_r P(TN, EI_n) = P_r \, P(TN, DN_1) \tag{77}$$

$$\frac{1}{N}\left[\frac{1}{P_r(1+K_{th})}\right] < P_T < \frac{2}{N}\left[\frac{1}{P_r(1+K_{th})}\right] \tag{78}$$

In questo modo, in condizioni di sincronismo sul target, sicuramente verrà generato uno spike in uscita.

In fase di progettazione, si dovranno seguire i seguenti step:

- Calcolo del $P(DN_1, EI_1)$ per ottenere condizioni di sincronismo sul target, a partire da τ_{in}
- Calcolo del P_T per ottenere l tolleranza desiderata (K_d è gia fissato)
- Calcolo del t_f minimo del target (in condizioni di sincronismo sul target, ovvero in assenza di errore sugli ingressi) :

Considerando una struttura a tre rami, si ottengono così le 4 formule finali:

$$P(DN_1, EI_1) = \frac{\left[\left\{\tau in_1 + \tau in_2 - \left[\frac{1}{Iext_1 - 1}\right] + \left[\frac{1}{Iext_3 - 1}\right]\right\}\right]}{P_r} \quad (79)$$

$$P(DN_2, EI_2) = \frac{\left[\left\{\tau in2 - \left[\frac{1}{Iext_2 - 1}\right] + \left[\frac{1}{Iext_3 - 1}\right]\right\}\right]}{P_r} \quad (80)$$

$$MaxTol = \frac{1 + K_{th}}{2\,K_d} \quad (81)$$

$$P_T = \frac{K_d\,Tol + 1 + K_{th}}{3P_r} \quad (82)$$

4.3.3 – Mono-input *spike-timing sequence detector*

Una limitazione delle strutture precedentemente viste, è quella di essere obbligati a dover fornire, in ingresso ad esse, impulsi su *n* differenti linee.

Verranno di seguito presentate dei *sequence detector* capaci di accettare impulsi sulla stessa linea di ingresso ed esibire *domini di attivazione* non banali. Tale caratteristica è resa possibile dall'impiego di altri fenomeni quali il *periodo refrattario* e l'uso di *neuroni inibitori*.

4.3.3.1 – Struttura con impiego di neurone inibitore

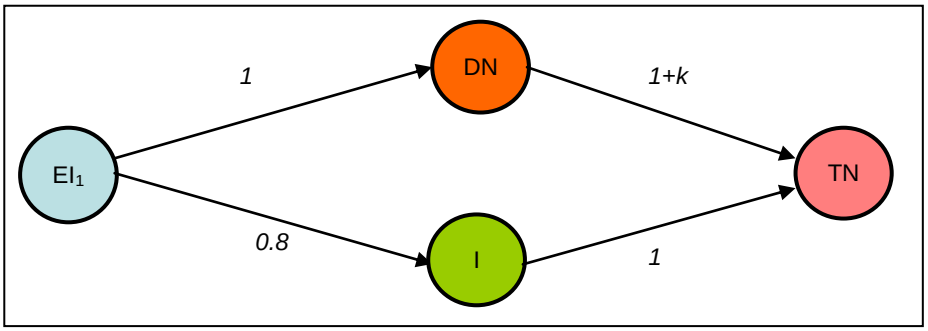


Fig.53: Struttura *Mono-input* spike-timing sequence detector, con neurone inibitore

- *DN* è di tipo *WMAL = 2*
- *I* è un inibitore di tipo *WMAL = 2*

Con tale configurazione è possibile ottenere la stessa tipologia di *dominio di attivazione* del caso delayed simple; l'effetto sarà ottenibile fornendo la sequenza da una sola linea di ingresso:

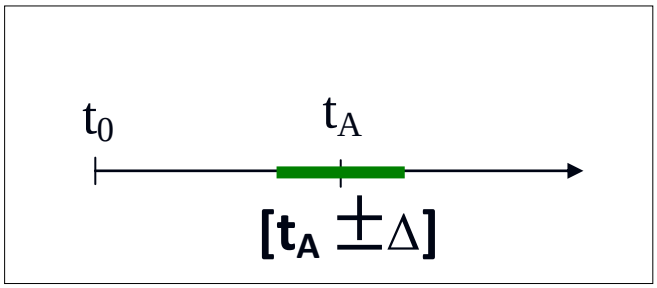


Fig.54: Dominio di attivazione per la struttura *Mono-input* spike-timing sequence detector, con neurone inibitore

4.3.3.2 – Struttura con impiego di neurone inibitore e tempo refrattario

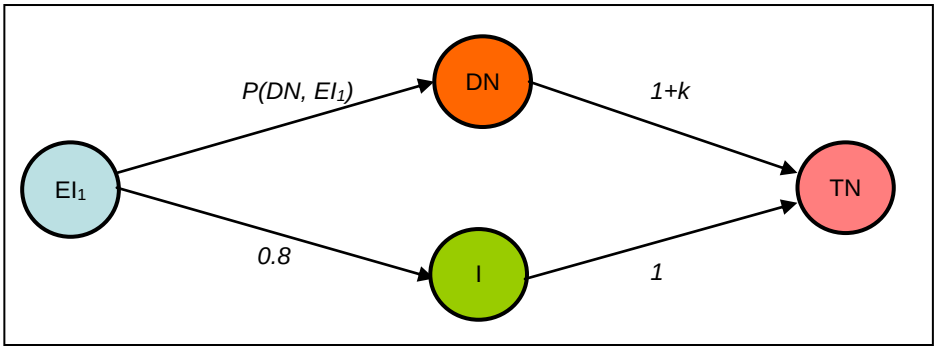


Fig.53: Struttura *Mono-input* spike-timing sequence detector, comprendente neurone inibitore e tempo refrattario.

- *I* è un inibitore di tipo *WMAL 2*

Al variare del livello *WMAL* di *DN* (1 o 2), si ottengono le due tipologie di dominio sotto raffigurate

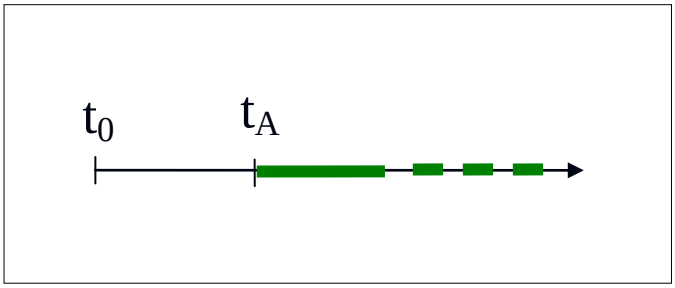


Fig.54: Dominio di attivazione per l'ultima struttura proposta: vediamo una nuova tipologia di dominio nel caso *DN* lev 1

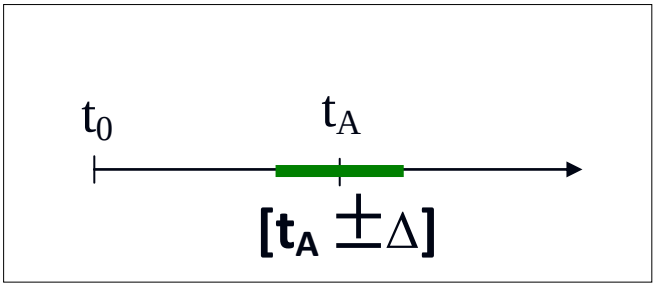


Fig.55: dominio nel caso *DN* di lev 2. Tale tipologia di dominio è stata gia visto per il delayed sequence… ma ottenibile da un ingresso solo.

4.4 – *Synchronism detector*

In questa sezione viene introdotta una struttura capace di rilevare sequenze di spike con una altissima precisione. Le proprietà della struttura proposta dipendono strettamente dalla presenza di neuroni inibitori, che sono in grado di conferire una maggiore sensibilità.

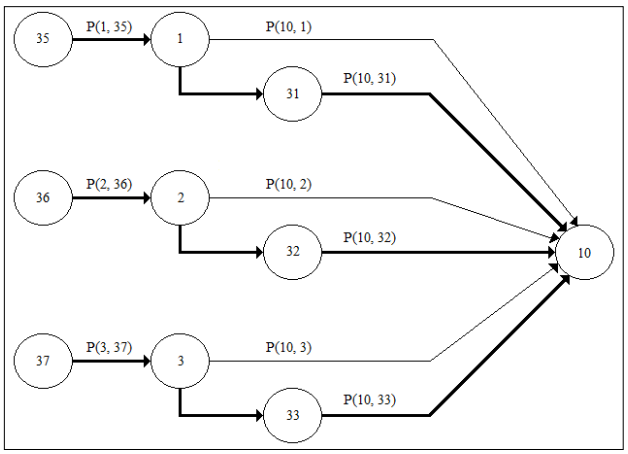


Fig.56. Synchronism detector. La rivelazione è positiva se il target neuron diviene attivo.

La struttura di esempio consiste in tre rami connessi ad un neurone target, anche se il numero di rami può essere esteso.
Nell'appendice sono riportati i risultati ottenuti dalle simulazioni fatte.

Le strutture funzionali descritte in in questo capitolo possono essere connesse ad un livello più alto, per eseguire specifici task.
Nel prossimo capitolo verranno presentate utili applicazioni ottenibili collegando opportunamente le strutture viste.

5. APPLICAZIONI E SVILUPPI FUTURI

Anche se con differenti approcci, in letteratura tecnica esistono vari lavori in cui sia stata applicata la *temporal pattern recognition* per applicazioni su reti neurali spiking [25],[26],[27]. Il campo di applicazione è paragonabile a quello delle tradizionali reti neurali sigmoidali e può essere suddiviso nelle stesse grandi aree:

- Auto-association.
- Pattern-association.
- Classificazione.
- Clustering.

Proprio per via del carattere dinamico delle SNNs, gran parte delle applicazioni sono impiegabili in aree concernenti la computazione di pattern temporali, come ad esempio il riconoscimento vocale e la predizione di andamenti vari [28]. Sulla base delle strutture elementari elencate precedentemente, possono essere comunque affrontati molti problemi di pattern recognition, come si vedrà nel corso di questo capitolo.

5.1 – Pattern recognition tramite Asynchronous SNN

Verrà considerato nella seguente trattazione un oggetto di riferimento descritto in base alle sue features, come nel classico approccio usato in problemi di pattern recognition con reti neurali tradizionali [29]. prescindendo dal particolare pre-processing operato, ogni feature verrà rappresentata in un dominio di riferimento e quindi oggetti diversi saranno mappati in uno spazio *n*-dimensionale; in questo modo si facilita di parecchio il problema della codifica neurale [31]: opportuni impulsi d'ingresso verranno definiti come input alla rete e l'informazione verrà codificata sottoforma di *spike-timings*.

Il riconoscimento delle classi di appartenenza sarà indicato dall'attivazione di neuroni target di riferimento. A livello neuronale, questo tipo di codifica è stato identificato in vari lavori, e prende il nome di *temporal coding*. Con l'approccio proposto si è quindi in grado di risolvere il problema con un meccanismo biologicamente plausibile.

Fig.57. Approccio proposto per un problema di classificazione, usando SNN.

A. Rappresentazione delle classi di riferimento tramite spike-timing.

Considerando un oggetto definito da una terna di features (spazio a 3 dimensioni), esso può essere rappresentato, dopo un processo di quantificazione delle proprie caratteristiche, in uno spazio delle features; si traccia la figura di seguito per una facilitata comprensione.

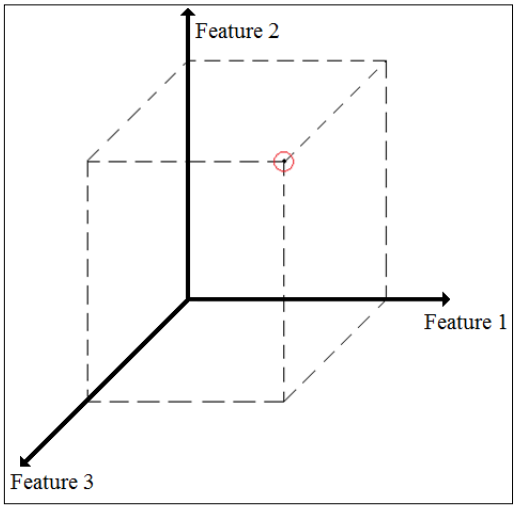


Fig.58: Un punto nello Spazio delle features. Ogni asse rappresenta un tempo.

A tal proposito, basterà creare un riconoscitore di sequenza per ogni feature dell'oggetto da descrivere e "accordare" i *domini di attivazione* delle strutture sull'intervallo che quantifica la descrizione della specifica caratteristica. In questo modo, set di riconoscitori, opportunamente dimensionati, potranno riconoscere le features descriventi determinate classi [30].

1° Esempio: oggetto descritto da una sola feature (figura 59). In questo caso basta un solo *sequence detector* per identificare la classe dell'oggetto. L'intervallo è centrato in un determinato istante e prevede una certa tolleranza.

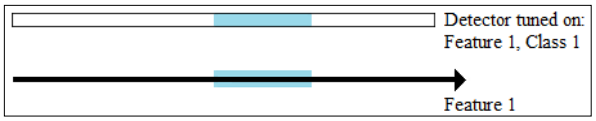


Fig.59: Caso di una singola classe, descritta da una singola feature. Il dominio di attivazione è accordato su una feature.

Ora, per distinguere oggetti appartenenti a più classi diverse si devono mappare tratti diversi sullo stesso asse, tramite diversi *sequence detectors*.

Tali riconoscitori lavorano in parallelo, col fine di attivare differenti neuroni target rappresentanti le possibili classi dell'oggetto.

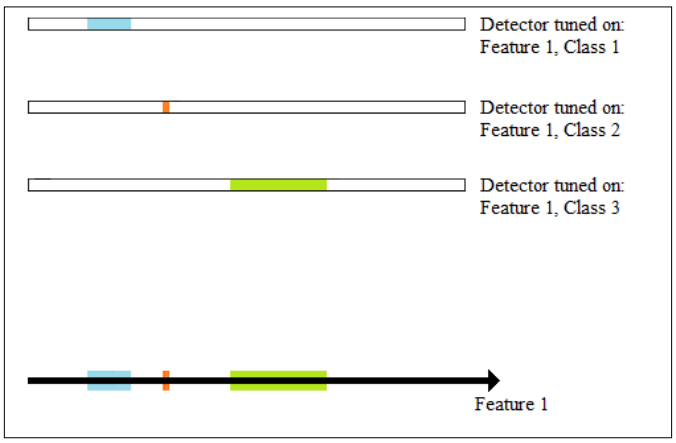


Fig.60: Caso di M classi, descritte da una singola feature. Si nota il dominio di attivazione parallelo.

2° Esempio: oggetto descritto da N features (per semplicità di rapprentazione si considera N=3). In questo caso ci sarà bisogno di un *set* di *N sequence detectors*, per identificare ogni classe. Nel particolare esempio, anche il numero delle classi è pari a tre.

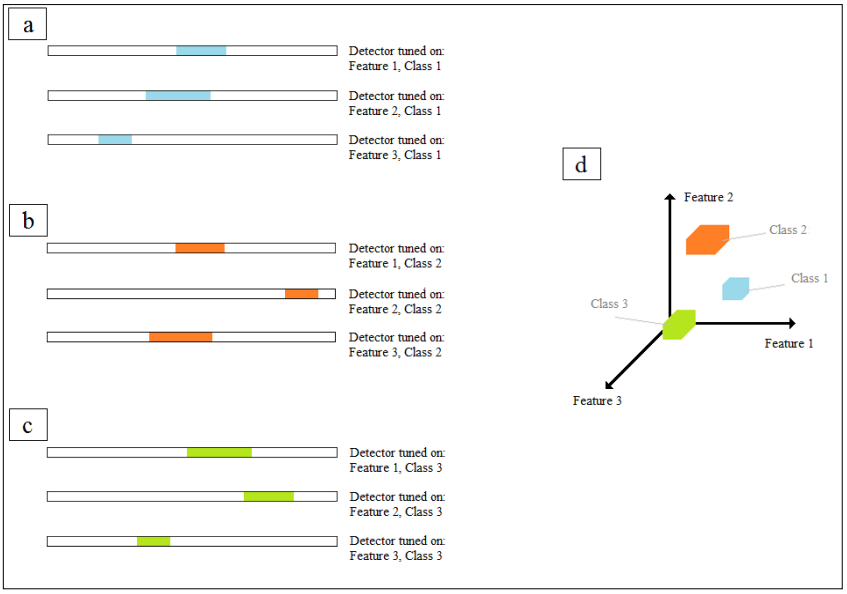


Fig.61: Caso di 3 feature, comprendenti tre classi ognuna: a) set di riconoscitori della classe 1; b) set di riconoscitori della classe 2; c) set di riconoscitori della classe 3; d) domini delle singole classi, nello spazio delle feature tridimensionale.

B. Architettura neurale di un classificatore Spike Timing based.

Lo spike *timings-based classifier* progettato, risulta così composto:

- uno *strato di ingresso*, in grado di ricevere informazione opportunamente codificata in termini di spike-timings. Il numero di IE è pari a *N+1*, ove *N* è il numero di features descriventi ogni classe.
- uno strato nascosto (hidden layer), ove avviene l'effettivo riconoscimento degli intervalli presentati alla rete, costituito da un numero *M* di *gruppi di riconoscimento*, pari al numero di classi da riconoscere.
- Uno strato di uscita, costituito dai neuroni target, ognuno dei quali si attiverà in corrispondenza dell'attivazione del relativo

gruppo di riconoscimento; anche questo strato è costituito da un numero di RG pari al numero *M* di classi da classificare.

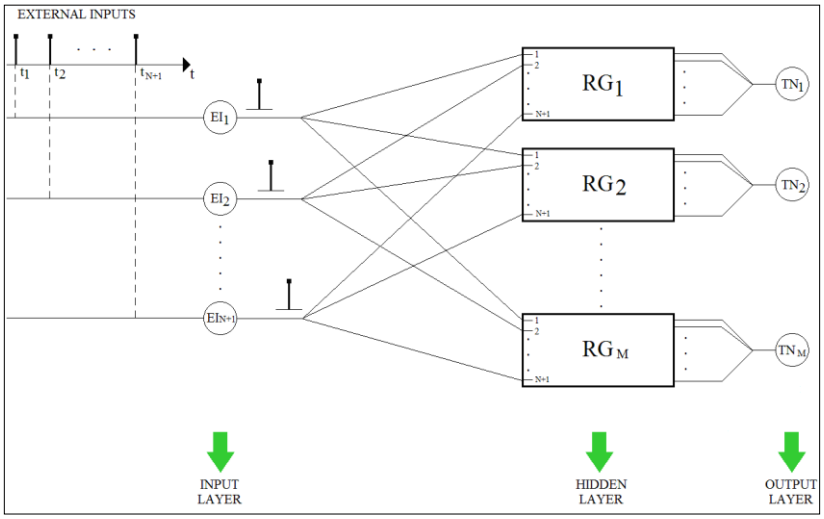


Fig.62: Architettura globale del classificatore proposto

Una possibile limitazione di questo approccio, sembra essere la forma dei domini di attivazione, che è sempre di tipo *right-angled*. In particolare, rettangolo nel caso di due feature, parallelepipedo per tre feature, iper-parallelepipedo per il caso generale. Questa limitazione può essere facilmente superata usando opportune "sottoclassi" implementati da differenti detectors, afferenti allo stesso target.

Uncaso di interesse nel problema di classificazione è quello di poter far aumentare-diminuire i *domini di attivazione* relativi alle classi identificate.

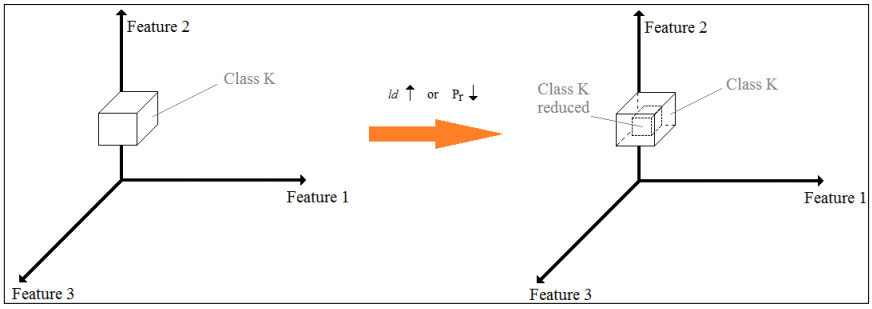


Fig.63: Incremento/decremento del volume di un dominio di attivazione.

Come prima mostrato, possono essere usati due parametri esterni*:* P_r e K_d. Tale possibilità risulta particolarmente utile quando differenti oggetti occupano domini di attivazione adiacenti; in questo caso ci sarebbe il problema della potenziale indeterminazione nella classificazione di tali oggetti. Se un ingresso dovesse attivare più di un target si potrebbe intervenire su dette grandezze in modo che venga selezionato un solo target. Al contrario, se nessuno dei target neurons venisse attivato da un pattern in ingresso, allora si potrebbe intervenire in maniera opposta maggiorando le zone di azione fino ad includere il punto rappresentante il pattern.

Ora è facile pensare a gruppi di sottoclassi adiacenti che siano implementati da diversi recognizer ma che facciano capo allo stesso neurone target (accorpamento delle *zone*). In questo modo il vincolo della forma ed accuratezza delle *zone* è superato.

C. Un esempio: il riconoscitore di quadri.

Sulla base delle strutture descritte, è ora possibile implementare opportuni set di riconoscitori per applicazioni reali. Tali "macro-

strutture" sono state connesse in modo che, dopo un opportuno training, siano in grado di riconoscere le caratteristiche relative ai quadri presentati.

Le opere utilizzate per dimostrare la funzionalità del classificatore sono di seguito elencate:

- *"L'Urlo", di Edvard Munch (1893)*

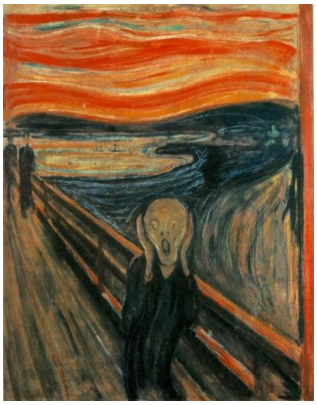


Fig.64: *L'urlo*

- *"La Gioconda", di Leonardo da Vinci (1503-1514)*

Fig.65: *La Gioconda*

- *"Notte Stellata", di Vincent Van Gogh (1889)*

Fig.66: *Notte stellata*

La scelta è ricaduta in particolare su queste tre opere per testare l'affidabilità della struttura con immagini che presentassero particolari caratteristiche sia a livello cromatico e di dimensioni: un buon banco di prova per la struttura soprattutto in termini di discriminazione, ove la stessa è chiamata a discriminare anche in caso ci siano delle differenze non così marcate tra le opere.

A tale scopo sono state considerate considerate le seguenti 5 feature:

- *Fattore di dinamica del colore Rosso;*
- *Fattore di dinamica del colore Verde;*
- *Fattore di dinamica del colore Blu;*
- *Fattore di forma, Shape,* legato alle dimensioni dell'immagine;
- Parametro *Edge* che descrive qualitativamente il contorno dell'immagine che si sta elaborando.

Sono state scelte appositamente features robuste a rotazioni e ridimensionamento. L'estrazione è avvenuta tramite un apposito script scritto in Matlab.

Il classificatore è stato realizzato utilizzando, per ciascuna feature, le strutture *spike timing delayed sequence detector* viste in precedenza, implementate in codice Matlab. Il dimensionamento dei pesi è stato fatto a partire dal *time-to-fire* per il valor medio, e la *deviazione standard* per la tolerance; tali valori sono stati calcolati prendendo in considerazione tutte le opere presenti nel training set.

Il *Detector Group*, costituito da cinque *sequence detector*, prende in ingresso un vettore con le features di un quadro: nella architettura realizzata in questo esempio ne esistono dunque 3, uno per opera. In uscita tali strutture sono collegate a neuroni esterni di target che operano una sommazione. Tali neuroni target hanno la particolarità di essere *senza perdita* (K_d nullo).

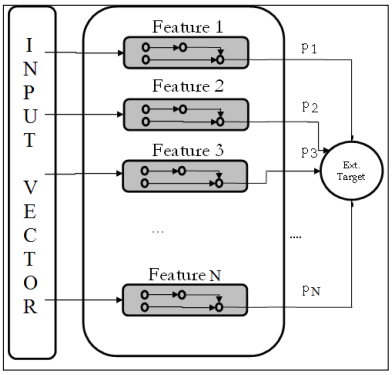


Fig.67: *Il detector group*

Continuando a salire con il livello di astrazione, si arriva a dare una rappresentazione del Classificatore vero e proprio; in questo caso è un

classificatore costituito solo da 3 Detector Group, dal momento che sono 3 le opere di cui è costituito il dataset:

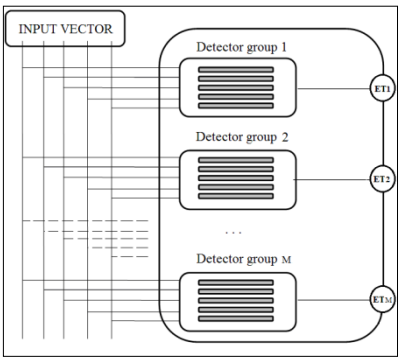


Fig.68: Il Classificatore.

Di seguito sono discussi i risultati ottenuti, utilizzando un dataset di 20 immagini per opera.

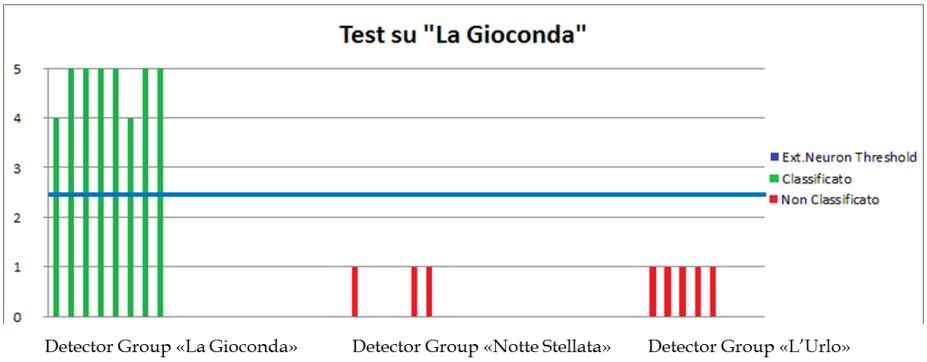


Fig.69: Grafico riportante i risultati della Struttura Classificatore "La Gioconda".

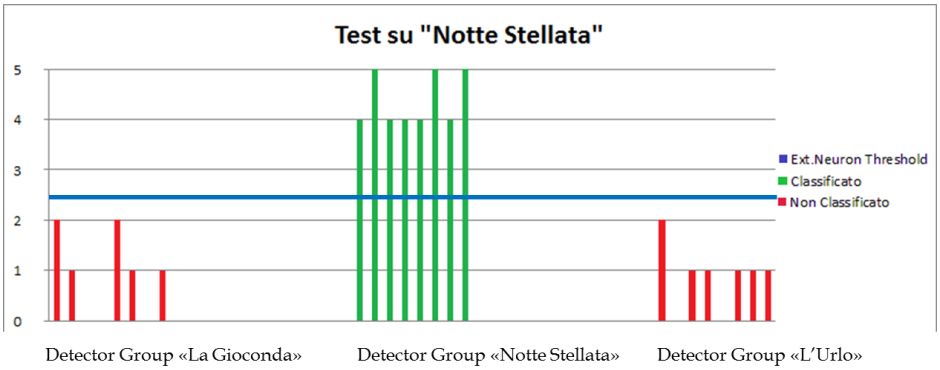


Fig.70: Grafico riportante i risultati della Struttura classificatore "Notte Stellata".

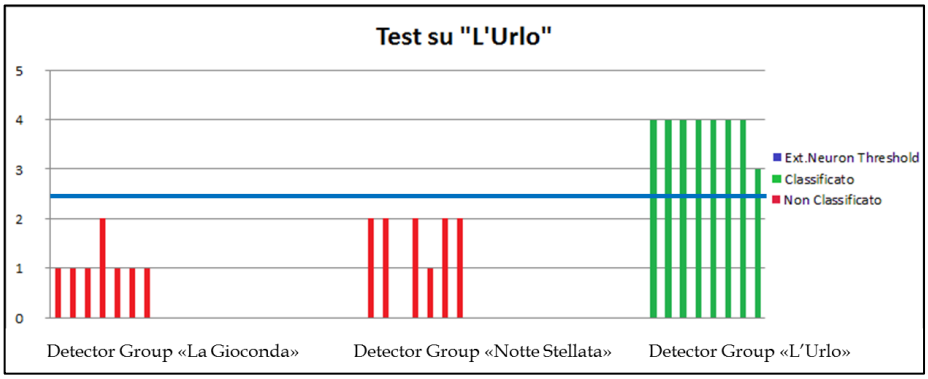


Fig.71: Grafico riportante i risultati della Struttura Classificatore "L'Urlo".

SNN CLASSIFIER						
	TP	**TN**	**FP**	**FN**	***Accuracy*** (%)	***Precision*** (%)
Detector Group "The Mona Lisa"	8	16	0	0	100	100
Detector Group "The Starry Night"	8	16	0	0	100	100
Detector Group "The Scream"	8	16	0	0	100	100

TAVOLA . Risultati ottenuti dalla classificazione

Dove *accuratezza* e *precisione* sono state determinate con le classiche formule:

$$Accuracy = (TP+TN)/(TP+FP+FN+TN) \quad (83)$$

$$Precision = TP/(TP+FP) \quad (84)$$

dove:

TN è il numero di veri negativi

TP è il numero di veri positivi

FN è il numero di falsi negativi

FP è il numero di falsi positivi

5.2 – Conclusioni

E' stato visto in questo capitolo come la connessione delle strutture presentate riesca a realizzare architetture molto più complesse, capaci di processare informazione in maniera parallela. Relativamente a ciò, è necessaria una trasformazione dei dati di ingresso dal dominio naturale a quello del tempo, in modo da poter fornire al classificatore una codifica temporale su cui poter lavorare.
Nell'esempio appena mostrato, l'ottimo esito delle prove di riconoscimento dipende sì dalla bontà delle features prese in considerazione e dal learning operato, ma dimostra che le strutture proposte sono perfettamente utilizzabili per la risoluzione dei classici problemi delle ANN, con un approccio sicuramente più biologicamente plausibile.

E' da tener presente, poi, che il modello globale realizzato, così come è stato visto nei capitoli precedenti, è in grado di mostrare proprietà particolari anche in evoluzione libera, a partire da un determinato stato globale. Ciò lascia immaginare la realizzabilità di applicazioni di tipo non supervisionato con il modello realizzato [32].

APPENDICE

In questa sezione sono riportati tutti i passaggi delle trattazioni di cui si fà riferimento nel testo. Per le simulazioni si fa riferimento al programma FIRNET, realizzato in ambiente Matlab dal Prof. M.Salerno.

A - *Multi branch direct spike-timing sequence detector:* prove di funzionamento.

La rete può essere caratterizzata tramite lo script "firnet_simple.m".

- External Input: 35, 36, 37
- Neuroni Eccitatori: 1 (sul primo ramo delay), 2 (sul secondo ramo delay)
- Neurone Target: 4

La rete nell'esempio ha un n=3,quindi $Iext_1$ è connesso al neurone 4 (target) per mezzo del neurone 1, la $Iext_2$ per mezzo del neurone 2, mentre la $Iext_3$ è connessa direttamente al neurone target.

Nello sviluppo di tale rete si deve tener conto sia della s*piking threshold* che del tempo di *linear decay*, che possono essere settati attraverso lo script "init.m" relativo al "firnet_simple.m" , e della *latency*. Questi tre parametri risultano fondamentali per stabilire le specifiche di progetto ed avere un corretto funzionamento di "RG".

- KL0=0.05; " linear decay " decadimento sottosoglia"
- KL1=0.04; " spiking threshold"
- La latency è definita direttamente dall'equazione del *time-to-fire* citata nel testo.

Si può osservare che il gruppo riconoscitore è costituito da soli neuroni eccitatori: il codice da valutare consisterà nella sequenza di intervalli forniti in ingresso al *recognizer group* (RG), caratterizzata da opportuni *interpulse intervals,* (intervalli tra un impulso e l'altro). "RG" sarà composto da percorsi pesati, in grado di elaborare la sequenza d'ingresso e fornire una risposta al neurone target.

Simulazioni

Il cuore della simulazione è il file "init.m", dove sono presenti le equazioni dedotte nel modello matematico della struttura, e da cui è possibile ricavare i pesi sinattici per caratterizzare la struttura:

```
    S(35,2)=0.00001;  %da qui possiamo impostare i tau
    S(36,2)=2;
    S(37,2)=5;

    num=3
    tol=0.5
    maxtol = (((1+KL1)/pre*(num-1))-
((1+KL1)/pre*(num)))/(-KL0)
    tau_tot=S(37,2)-S(35,2)
    tau_1=S(36,2)-S(35,2)
    tau_2=S(37,2)-S(36,2)
    Sa=1+(1/tau_tot)
    Sb=1+(1/((Sa-1)^(-1)-tau_1))
    Sc=((1+KL1)+tol*KL0)/(num*pre)
```

Si ottengono i seguenti valori:

```
    Pw(1,35)=1.2000;
    Pw(4,1)=0.3550;
    Pw(2,36)=1.3333;
    Pw(4,2)=0.3550;
    Pw(4,37)=0.3550;
```

In questo caso lanciando il file "firnet_simple.m" risulta:

```
nspt =
    35    36    37     1     2     4
>> dspt
dspt =
    0.0000    2.0000    5.0000    5.0000    5.0003
20.3885
S(4,1)
ans =
    1.0650
```

Come si vede, il neurone target è in spike.

Di seguito viene effettuata una variazione, all'interno della tolleranza: in questo caso viene modificato solamente il momento in cui agisce $Iext_2$, aumentandolo a 2.48 (spostando l'evento al limite della tolleranza di 0.5). I risultati sono i seguenti:

```
nspt =
    35    36    37     1     2     4
dspt =
    0.0000    2.4800    5.0000    5.0000    5.4803
29.8795
S(4,1)
ans =
    1.0410
```

Come si può osservare, lo stato del neurone target è diminuito, rispetto al caso precedente, arrivando al limite dello spike (latenza massima). Nel caso seguente viene invece variato il momento in cui agisce $Iext_2$, portando l'evento al limite inferiore della tolleranza (1.52)

```
nspt =
    35    36     2    37     1     4
>> dspt
dspt =
    0.0000     1.5200     4.5203     5.0000     5.0000
29.3816
S(4,1)
ans =
    1.0410
```

Il risultato appare simmetrico al caso precedente. Ora viene mostrato il comportamento della struttura se l'evento occorre in un istante non coperto dalla tolleranza (2.51)

```
nspt =
    35    36    37     1     2
>> dspt
dspt =
    0.0000    2.5100    5.0000    5.0000    5.5103
S(4,1)
ans =
    1.0395
```

Si nota che il neurone target non è in spike.

Con ciò è dimostrato che, al di fuori della tolleranza sia per eccesso che per difetto, la struttura non funziona.

I risultati di cui sopra sono stati ottenuti variando solamente l'ingresso $Iext_2$.

B - Trattazione completa del *simple delayed spike-timing sequence detector*

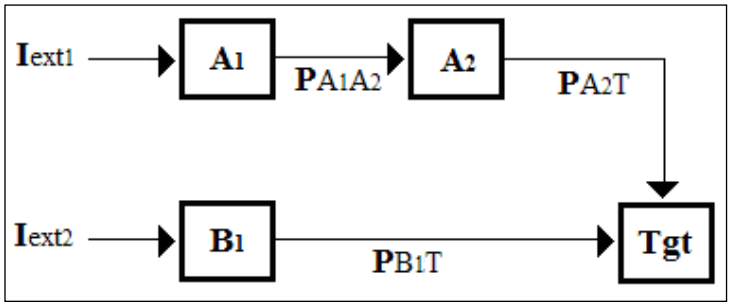


Fig. B1: Struttura del simple delayed spike timing sequence detector

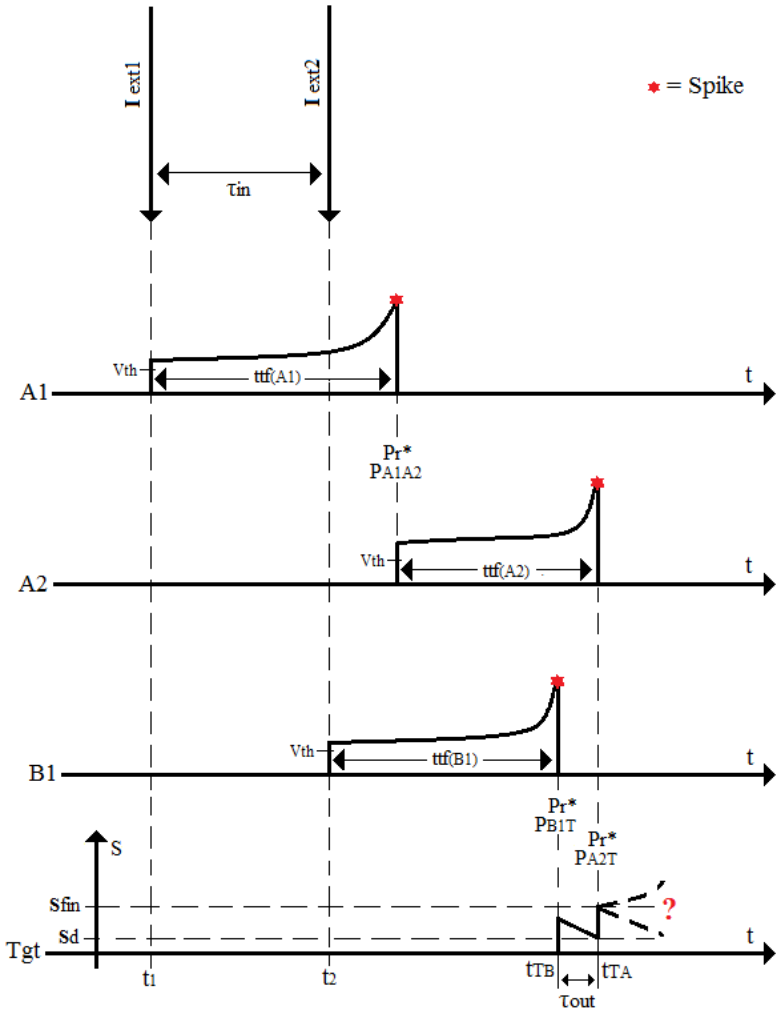


Fig. B2: diagramma temporale della struttura proposta

I requisiti necessari per il corretto funzionamento della struttura sono i seguenti:

- A_1: Level 1 working mode
- A_2: Level 1 working mode
- B_1: Level 1 working mode
- *TN:* Level 2 working mode

quindi:

- per assicurare l'attivazione dei neuroni dello strato di ingresso (A_1 e B_1), deve risultare:

$$Iext_1 \ \& \ Iext_2 > 1 + K_{th}$$

$$P_{A1A2} > 1 + K_{th}$$

- per assicurare l'attivazione del neurone A2 deve risultare:

$$P_r \ P_{A1A2} > 1 + Kth \rightarrow P_{A1A2} > (1 + K_{th}) / P_r$$

- per assicurare lo spike in uscita, nella favorevole condizione di sincronismo sul target, dev'essere:

$$P_r \ (P_{B1T} + P_{A2T}) > 1 + K_{th} \rightarrow P_{B1T} + P_{A2T} > (1 + K_{th}) / P_r$$

Analisi (vedere il grafico della struttura per i riferimenti alle grandezze citate):

- $ttf(A_1) = 1/(Iext_1 - 1)$
- $ttf(A_2) = 1/(Pr\ P_{A1A2} - 1)$
- $ttf(B_1) = 1/(Iext_2 - 1)$
- $t_{TA} = ttf(A_1) + ttf(A_2) + t_1$
- $t_{TB} = ttf(B_1) + t_2 = t_1 + \tau_{in} + ttf(B_1)$

Quindi al target risulta:

- nel caso in cui al target arriva prima il contributo da B_1 e poi quello da A_2:
  $S_d(B_1,A_2) = P_r\, P_{B1T} - K_d\ \tau_{out}(B_1,A_2) = Pr\, P_{B1T} - K_d\,(t_{TA} - t_{TB})$;

- nel caso in cui al target arriva prima il contributo da A_2 e poi quello da B_1:
  $S_d(A_2,B_1) = P_r\, P_{A2T} - K_d\ \tau_{out}(A_2,B_1) = Pr\, P_{A2T} - K_d\,(t_{TB} - t_{TA})$;

allora, indifferentemente da quale dei due contributi arrivi prima al target, si ha:

$$S_{fin} = P_r\, P_{B1T} + Pr\, P_{A2T} - K_d\,|\,t_{TB} - t_{TA}\,|$$

nella ultima relazione è stato inserito l'operatore *modulo*, in modo da generalizzare la formula, indifferentemente dall'ordine di arrivo al target dei contributi provenienti da B_1 ed A_2.
La condizione di spike al target risulta:

$$S_{fin} > (1 + K_{th})$$

Andando a sostituire con le relazioni appena espresse, si trova:

$$P_r\, P_{B1T} + Pr\, P_{A2T} - K_d\ |\,t_{TB} - t_{TA}\,| > (1 + K_{th})$$

Esplicitando i valori di t_{TA} e t_{TB} precedentemente trovati, si ottiene:

$$P_r\, P_{B1T} + P_r\, P_{A2T} - K_d\,|ttf(B_1) + t_2 - [ttf(A_1) + ttf(A_2) + t_1)]| > (1 + K_{th})$$

cioè:

$$Pr\,(P_{B1T} + P_{A2T}) - K_d\,|[1/(Iext_2 - 1)] - [1/(Iext_1 - 1)] - [1/(P_r\, P_{A1A2} - 1)] + \tau in)| > (1 + K_{th})$$

C - Trattazione completa del *telescopic delayed spike-timing sequence detector*

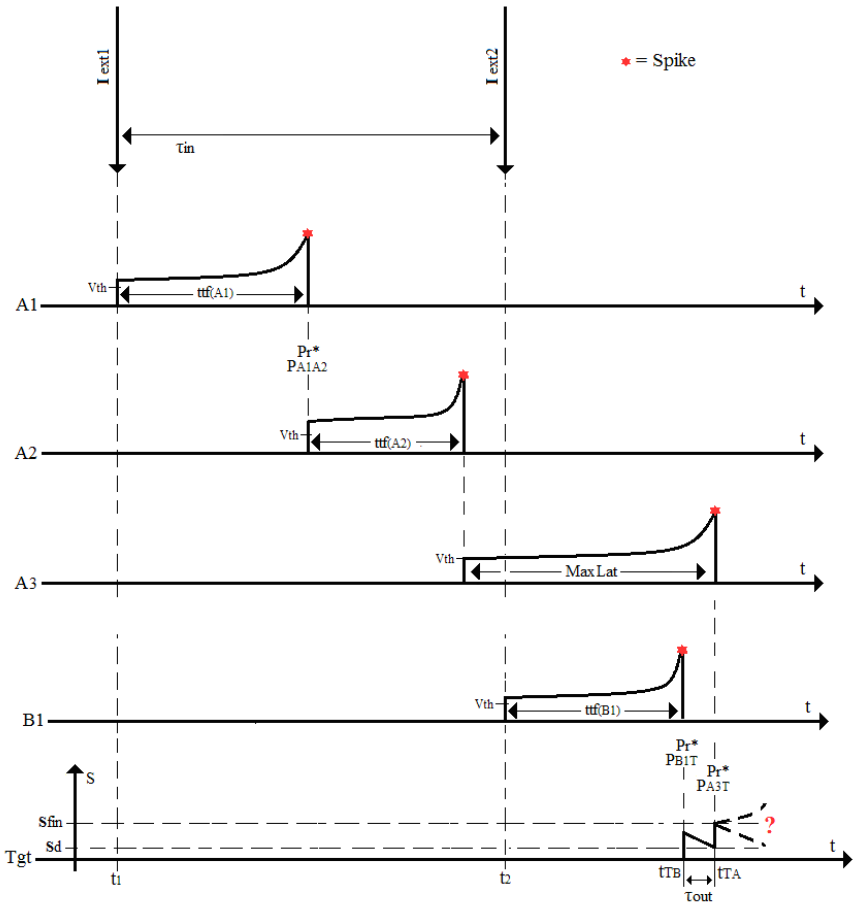


Fig. C1: caso **N=2** (struttura delayed con un neurone in più)

Progettazione (vedere il grafico precedente per i riferimenti alle grandezze citate):

- Calcolo preliminare del *numero di neuroni necessari nel ramo superiore* (N_{delay_line}), in base al τin:

  $$N_{delay\_line} = tr\ [(\tau_{in} + ttf(B_1) + Tol - ttf(A_1)) / MaxLat\ ] + 1$$

  ove *tr* rappresenta l'operazione di *troncamento*, in quanto serve un numero intero per individuare una quantità di neuroni.
- Ipotizzando che, tranne il primo del ramo, gli altri siano settati con i pesi:

$$P(n\text{-}1,n) = 1/\{[(1+K_{th})\text{ -}1]P_r\}$$

Per ottenere la condizione di sincronismo sul target, si calcola PA1A2 a partire da τ_{in}:

$$\tau_{in} = ttf(A_1) + ttf(A_2) + N_{delay\_line}\, MaxLat - ttf(B_1) - \tau_{out}$$

con $\tau_{out} = 0$, cioè:

$$\tau_{in} = ttf(A_1) + ttf(A_2) + (N_{delay\_line} - 1)\, MaxLat - ttf(B_1)$$

$$\tau_{in} = ttf(A_1) + ttf(A_2) + (N_{delay\_line} - 1)\,\{1/[(1+ K_{th}) - 1]\} - ttf(B_1)$$

$$\tau_{in} = [1/(Iext_1 - 1)] + [1/(P_r\, P_{A1A2} - 1)] + (N_{delay\_line} - 1)\,\{1/[(1+ K_{th}) -1]\} - [1/(Iext_2 - 1)]$$

$$\tau_{in} - [1/(Iext_1 - 1)] + [1/(Iext_2 - 1)] - (N_{delay\_line} - 1)\,\{1/[(1+K_{th})-1]\} = [1/(P_r\, P_{A1A2} - 1)]$$

$$\{\tau_{in} - [1/(Iext_1 - 1)] + [1/(Iext_2 - 1)] - (N_{delay\_line} -1)\,\{1/[(1+ K_{th}) - 1]\}\}^{-1} = (P_r\, P_{A1A2} - 1)$$

$$P_{A1A2} = [\{\tau_{in} - [1/(Iext_1 - 1)] + [1/(Iext_2 - 1)] - (N_{delay\_line} - 1)\,[1/(1+K_{th} -1)]\}^{-1} +1] / P_r =$$

$$[\{\tau_{in} - [1/(Iext_1 - 1)] + [1/(Iext_2 - 1)] - (N_{delay\_line} - 1)\,(1/ K_{th})\}^{-1} +1] / P_r$$

(caso generico *telescopico*, agevolmente riconducibile a struttura base)

- Calcolo P_T per ottenere la tolleranza desiderata (K_d è gia fissato):

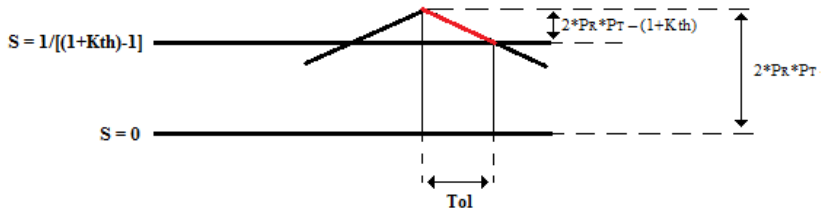


Fig. C2: Sincronismo nella condizione di sommazione contributi al Target

se i contributi arrivassero in maniera sincrona, cioè in assenza di errore, il valore dello stato raggiunto dal target sarebbe pari a $2P_rP_T$;

se i contributi arrivasseroo al massimo scostamento possibile, definito con la Tolerance (in uscita c'è una mera traslazione), il valore dello stato raggiunto dal Target è pari a $(1+K_{th})$.
Ora si cerca una relazione per ricavare P_T, in funzione della tolleranza desiderata; prendendo in considerazione il segmento rosso in figura, si ha:

$$m\ (coefficiente\ angolare) = -\ K_d = (y_2-y_1)/(x_2-x_1)$$

cioè:

$$-\ K_d\ = \{0-[2P_r\,P_T - (1+K_{th})] / [Tol-0] = -\ (2P_r\,P_T - 1 - K_{th}) / Tol$$

da cui

$$K_d\ =\ (2P_rP_T\ -\ 1\ -\ K_{th})\,/\,Tol$$
$$P_T\ =\ (K_d\,Tol\ +\ 1\ +\ K_{th})\,/\,2P_r$$

- Adesso si può trovare il *ttf* minimo del *TN* (in condizioni di sincronismo sul target, ovvero in assenza di errore) :

$$ttf_{min}(TN)\ =\ 1/(\ 2P_rP_T\ -\ 1)$$

- E' ora possibile calcolare l'intervallo di tempo che intercorrerebbe tra il secondo impulso e l'"accensione" del target, in condizioni di assenza di errore:

$$ttf_{B1}\ +\ ttf_{min}\,(N)$$

D - Simulazioni effettuate con la struttura *synchronism detector*

TABLE I

n	t
35	7.0000
37	7.0000
36	7.0000
1	17.0000
2	17.0000
3	17.0000
31	18.9231
33	18.9231
32	18.9231
10	19.9231

TABLE II

n	t
35	7.0000
37	7.0000
36	7.0000
1	17.0000
2	17.0000
3	17.0000
31	18.9231
33	18.9231
32	18.9331
--	---

TABLE III

n	t
35	7.0000
37	7.0000
36	7.0000
3	17.0000
1	17.0000
33	17.0000
31	18.9231
2	18.9231
32	18.9231
--	---

TABLE IV

n	t
36	7.0000
35	15.0000
37	15.5714
3	17.0000
2	17.0000
1	17.0000
33	18.9230
32	18.9231
31	18.9231
10	19.9256

Table D I: Firing table for the case A simulation. The detection is positive.
Table D II:Firing table for the case B simulation. No firing for the neuron TN_{10}. The detection is negative.
Table D III: Firing table for the case C simulation. No firing for the neuron TN_{10}. The detection is negative.
Table D IV: Firing table for the case D simulation. The detection is positive.

- External Inputs : neurons 35, 36, 37
- Excitatory Neurons : neurons 1, 2, 3
- Inhibitory Neurons : neurons 31, 32, 33
- Target Neuron : neuron 10

$P(1,35) = P(2,36) = P(3,37) = 1.1$
$P(31,1) = P(32,2) = P(33,3) = 1.52$
$P(10,1) = P(10,2) = P(10,3) = 0.5$
$P(10,31) = P(10,32) = P(10,33) = -4$

$P(1,35) = 1.5$; $P(2,36) = 1.1$; $P(3,37) = 1.7$

$t(1) = 9.0000$, $t(2) = 17.000$, $t(3) = 8.4286$

RINGRAZIAMENTI

Sono del parere che lavorare per quello che si ama è come giocare... anche se questa è stata un'esperienza dura, mi ritengo privilegiato perchè ho avuto la possibilità di mischiare il lavoro alle mie passioni; spero adesso di avere l'opportunità di proseguire il mio percorso in ambito universitario.

Ringrazio la mia famiglia e i miei amici, Alicia, il Prof. Mario Salerno ed il Prof. Marco Re.

Ringrazio i 3/4 della *Seeinteracting crew*, di cui faccio parte: Maurizio Massarelli, Marco Bertola, Emiliano Daddario, coi quali spero di realizzare presto qualcosa di importante.

Ringrazio i validi tesisti, ognuno dei quali ha contribuito a realizzare un pezzo del puzzle, specialmente Alessandro Cristini (oggi caro amico e valido collega), Yari Sanfelice e Andrea D'Annessa.

Ringrazio inoltre i colleghi, gli studenti e i collaboratori del Master in Ingegneria del Suono di Tor Vergata.

Grazie a tutti per essermi stati vicini durante lo svolgimento di questo lavoro.

Gianluca

Bibliografia

1. "Networks of spiking neurons: The third generation of neural network models", W. Maass; *Neural Networks*, 1997

2. "Which Model to Use for Cortical Spiking Neurons?", E. M. Izhikevich; *IEEE Transactions on Neural Networks*, 2004

3. "Spike-timing dynamics of neuronal groups", E. M. Izhikevich, J. A. Gally, and G. M. Edelman: *Cerebral Cortex*, 2004

4. "Random Walk Models for the Spike Activity of a single neuron", G. L. Gerstein, B. Mandelbrot; *Biophysical Journal*, 1964

5. "A review of the integrate-and-fire neuron model: I. Homogeneous synaptic input", A. Burkitt; *Biological Cybernetics*, 2006

6. "A review of the integrate-and-fire neuron model: II. Inhomogeneous synaptic input and network properties", A. Burkitt; *Biological Cybernetics*, 2006

7. "Polychronization: Computation with spikes," E. M. Izhikevich; *Neural Computation*, 2006.

8. "Release-Dependent Variations in Synaptic Latency: A Putative Code for Short- and Long-Term Synaptic Dynamics" S. Boudkkazi, E. Carlier, N. Ankri, et al.; *Neuron*, 2007

9. "Dynamical system in neuroscience: the geometry of excitability and bursting", E. M. Izhikevich; *MIT Press*, 2007.

10. "Simulation of networks of spiking neurons: A review of tools and strategies", R. Brette, M. Rudolph, T. Carnevale, et al.; *Journal of Computational Neuroscience*, 2007.

11. "Accurate Latency Characterization for Very Large Asynchronous Spiking Neural Networks", M. Salerno, G. Susi, A. Cristini; *Proceedings of the fourth International Conference on Bioinformatics Models, Methods and Algorithms*, 2011

12. "Neural Darwinism: The Theory of Neuronal Group Selection", M. Edelman; *New York: Basic Book*, Inc., 1987

13. "Spiking neural networks as analog dynamical systems: basic paradigm and simple applications", M.Salerno, G.Susi, A.D'Annessa, A.Cristini, Y. Sanfelice; *International advances on Computer Engineering*, 2012

14. "A logical calculus of the ideas immanent in nervous activity", W.S.McCulloch and W.Pitts; *bulletin of Mathematical Biophysics*, 1943

15. "The perceptron: A probabilistic model for information storage and organization in the brain", F.Rosenblatt; *Psychological Review*, 1958

16. "Pulsed Neural Networks", W.Maass, C.M.Bishop; *Maass and Bishop editors*, 2001

17. "Evolving Spiking Neural Network Controllers for Autonomous Robots", Hani Hagras, Anthony Pounds-Cornish, Martin Colley, Victor Callaghan and Graham Clarke; *Proceedings of the 2004 IEEE International Conference on Robotics and Automation*, 2004

18. *"NEURON software"*, simulatore disponibile qui*: http://www.neuron.yale.edu/neuron/*

19. "Accelerating Event-Driven Simulation of Spiking Neurons with Multiple Synaptic Time Constants", D'Haene, B. Schrauwen, J. V. Campenhout and D. Stroobandt; *Neural Computation*, 2009.

20. "Principi di neuroscienze", E.R.Kandel, J.H.Schwartz, J.M.Thomas; *CEA*, 2003

21. "A quantitative description of membrane current and application to conduction and excitation in nerve", A. L. Hodgkin, A. F. Huxley; *Journal of Physiology*, 1952

22. "Mathematical models of threshold phenomena in the nerve membrane", R. FitzHugh; *Bull. Math. Biophys*, 1955

23. "Efficient simulation scheme for spiking neural networks", R.R.Carrillo Sánchez; PhD Dissertation, 2008

24. "Introduction to spiking neural networks: Information processing, learning and applications", F.Ponulak, A.Kasiński; *Acta Neurobiol*, 2011

25. "Advances in design and application of spiking neural networks", A. Belatreche, L. P. Maguire, M. McGinnity; *Soft Computing - A Fusion of Foundations, Methodologies and Applications*, 2007

26. "Computing with Spiking Neuron Networks", H.Paugam-Moisy , S.M. Bohte; *Handbook of Natural Computing*, 2009

27. "Temporal pattern recognition using spiking neural networks for cortical neuronal spike train decoding", H.Fang, Y.Wang, J.He, S.Liu; *Neural Computation (MIT Press)*, 2008

28. "Spiking neural networks: a survey", H.P.Moisy; *IDIAP research report*, 2006

29. "Comparison of Different Classifiers on a reduced set of features for Mental Tasks-Based Brain-Computer Interface", Giovanni Saggio, Pietro Cavallo, Lucia Rita Quitadamo, Maria Grazia Marciani, Luigi Bianchi,Giovanni Costantini, Gianluca Susi; *Proceedings of the Third International Conference on Bio-inspired Systems and Signal Processing*, 2010

30. "Spiking neural networks as continuous-time dynamical systems: fundamentals, elementary structures and simple applications", Mario Salerno, Gianluca Susi, Alessandro Cristini, Yari Sanfelice, and Andrea D'Annessa; *International journal of information technology: extended versions*, 2012

31. "Implementazione ed analisi di reti neurali wetware", G.Cino; PhD Dissertation, 2007

32. "Unsupervised Classifcation of Complex Clusters in Networks of Spiking Neurons", S.M. Bohte, J.N. Kok, H.La Poutre; *NSF*, 2000

ALTRI TESTI e SITI CONSULTATI:

"A Discrete-Event Neural Network Simulator for General Neuron Models" Makino, T.; Neural Comput & Applic., 2003.

"A pulse-coded communications infrastructure for neuromorphic systems", S.R. Deiss, R.J. Douglas, A.M. Whatley in: W. Maass, C.M. Bishop (Eds.), Pulsed Neural Networks; MIT Press, Cambridge, 1998.

"Cellular Neural Networks: Theory", L.O.Chua, L.Yang; IEEE Trans. Circuits Syst., 1988.

"Communicating neuronal ensembles between neuromorphic chips", K.A. Boahen, T.S. Lande (Ed.); Neuromorphic Systems Engineering, Kluwer Academic Press, Norwell, 1998.

"Competitive Hebbian learning through spiketiming dependent synaptic plasticity" Song S, Miller K and Abbott LF;Nature Neuroscience, 2000.

"Computer model of antiepileptic effects mediated by alterations in GABAA mediated inhibition", Thomas E and Lytton W; NeuroReport, 1997.

"Dynamics of encoding in a population of neurons", Knight, B.W., J. Gen. Physiol, 1972.

"Efficient modelling of spiking neural networks on a scalable chip multiprocessor", Xin Jin; Furber, S. B.; Woods, J. V; IEEE International Joint Conference on Neural Networks (IEEE World Congress on Computational Intelligence), 2008.

"Encoding of categories by non-category specific neurons in the inferior temporal cortex", Thomas E., Van Hulle M. & Vogels R; Journal of Cognitive Neuroscience, 2000

"Increased synchrony with increase of a low threshold calcium conductance in a model thalamic network: a phase shift mechanism", Thomas E and Grisar T ; Neural Computation, 2000.

"Independent Variable Time-Step Integration of Individual Neurons for Network Simulations", Lytton, W. W. and M. L. Hines; Neural Comp., 2005.

"Modeling thalamocortical oscillations", Lytton W and Thomas E; cerebral Cortex. Volume 13, 1997

"Neurons with graded response have collective computational properties like those of two-state neurons", Hopfield, J. J; Proc. Natl. Acad. Sci., 1984.

"Numerical recipes in C: The art of scientific computing ", W. H., Flannery, B. P., Teukolsky, S. A., & Vetterling,W. T; Cambridge University Press, 1988.

"Relationship between the firing rate of a single neuron and the level of activity in a population of neurons. Experimental evidence for resonant enhancement in the population response." Knight, B.W. The J. Gen. Physiol. 1972.

"Repetitive impulse discharge of a simple model compared to that of spinal motoneurones. ", Kernell, D.; The Brain Research, 1968.

"Shunting inhibition does not have a divisive effect on firing rates", Holt, G. R.,&Koch, C., Neural Comput., 1997.

"The double queue method: a numerical method for integrate-and-fire neuron networks", Lee, G. and N. H. Farhat; Neural Networks, 2001.

"The neural code between neocortical pyramidal neurons depends on neurotransmitter release probability", Tsodyks MV and Markram H Proc Natl Acad Sci 94, 1997.

"A quantitative description of membrane current and application to conduction and excitation in nerve", A.L. Hodgkin, A.F. Huxley; Journal of Physiology, 1952.

"Accelerating Event-Driven Simulation of Spiking Neurons with Multiple Synaptic Time Constants", M. D'Haene, B. Schrauwen, J.V. Campenhout and D. Stroobandt; Neural computation, 2009.

"Histologie du Systeme Nerveux de l'Homme et des Vertebres, vol. I & II", S. Ramon y Cajal, 1909, 1911.

"Random walk models for the spike activity of a single neuron", G.L. Gernstein, B. Mandelbrot. Biophisical journal, 1964

"Recherches quantitatives sur l'excitation électrique des nerfs traitée comme une polarization", L. Lapicque, 1907.

"Toward an integrated continuum model of cerebral dynamics: the cerebral rhythms, synchronous oscillation and cortical stability", Wright JJ, Robinson PA, Rennie CJ, Gordon E, Bourke PD, Chapman CL, Hawthorn N, Lees GJ, Alexander D.; Biosystems. 2001.

http://www.elastyc.unimore.it/fonda/DISPENSA_Tb_html/Programma/Prog_TB_1/Hodg-Hux.htm