# INSTITUTO POLITÉCNICO NACIONAL

## CENTRO DE INVESTIGACIÓN EN COMPUTACIÓN

# MEMORIAS ASOCIATIVAS BIDIRECCIONALES ALFA-BETA

T E S I S

QUE PARA OBTENER EL GRADO DE

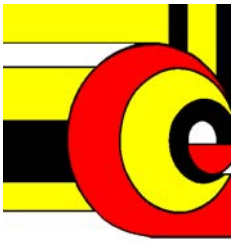
## DOCTOR EN CIENCIAS DE LA COMPUTACIÓN

PRESENTA:

MARÍA ELENA ACEVEDO MOSQUEDA

DIRECTOR DE TESIS:

**DR. CORNELIO YÁÑEZ MÁRQUEZ**



MÉXICO, D.F. JULIO DE 2006

# *DEDICATORIA*

Este trabajo lo dedico a quienes siempre me han apoyado incondicionalmente:

A María Elena y Crescencio, mis padres.

A Juan y Antonio, mis hermanos.

A Sebastián y Mariana, mis dos rayitos de luz.

A ese ser muy especial, el sol de todas mis mañanas, mi único amor fiel y sincero, mi constante motivación.

*Las grandes mentes discuten ideas; mentes promedio discuten los eventos; mentes pequeñas discuten a las personas.*

**Anónimo**

# *AGRADECIMIENTOS*

Un agradecimiento muy especial al Dr. Cornelio Yáñez por compartir conmigo sus conocimientos, por brindarme su apoyo y paciencia; porque su impecable desempeño académico ha sido la motivación para mejorar día con día mi trabajo. Dr. Yáñez mi respeto y admiración para usted.

A Julia y Vicente, mis dos grandes amigos, por estar conmigo en los momentos difíciles, por su comprensión, sus consejos y ese gran cariño que me han regalado. Gracias por enseñarme el significado de la palabra amistad.
Julia, Vicente: los quiero mucho.

A Ade por las facilidades brindadas para el material bibliográfico necesario a lo largo de mis estudios de doctorado. Principalmente, gracias por tu amistad de tantos años.

A mis sinodales: Dr. Suárez, Dr. Pogrebnyak, Dr. López Bonilla y Dr. Pastor, por todo el apoyo y el interés mostrado hacia mi trabajo. Sus conocimientos y opiniones fueron muy importantes para alcanzar esta meta.

Gracias a todas las personas que, de alguna u otra forma, contribuyeron para que este gran sueño ahora sea realidad.

# Índice General

# Índice de Tablas

# Índice de Figuras

# Resumen

En este trabajo se presenta un nuevo modelo de Memorias Asociativas Bidireccionales (BAM, *Bidirectional Associative Memories*) llamado BAM Alfa-Beta. El modelo está basado en las Memorias Asociativas Alfa-Beta de las cuales hereda el nombre.

Debido a su origen, la BAM Alfa-Beta no requiere de un proceso iterativo de convergencia hacia estados estables, a diferencia de los modelos basados en la BAM de Kosko; por la misma causa, no tiene problemas de estabilidad.

Los patrones que utiliza la BAM Alfa-Beta son binarios.

La capacidad de recuperación del modelo propuesto es de $2^{\min(n,m)}$, siendo $n$ y $m$ las dimensiones de los patrones de entrada y salida, respectivamente. Entonces, la BAM Alfa-Beta siempre recupera de forma correcta todos los patrones entrenados. La recuperación correcta no requiere de ninguna condición previa de las propiedades de los patrones, como: distancia de Hamming, ortogonalidad o separación lineal.

Se presenta el fundamento matemático que sustenta el por qué la BAM-Alfa siempre tendrá recuperación correcta.

Se proponen dos nuevas transformadas vectoriales: de expansión y de contracción.

En la fase de aprendizaje, la BAM Alfa-Beta hace uso del código binario *one-hot* y de un nuevo código propuesto: el código binario *zero-hot*.

La BAM Alfa-Beta consta de 4 etapas, dos etapas se utilizan en el sentido hacia delante, y las dos etapas restantes corresponden al sentido hacia atrás, lo que permite la bidireccionalidad del modelo.

La primera etapa, en el sentido hacia delante, esta conformada con dos memorias autoasociativas Alfa-Beta, *max* y *min*, construidas a partir de los patrones de entrada. En la segunda etapa se construye un *Linear Associator* modificado a partir de los patrones de salida. La tarea de la primera etapa es obtener un vector *one-hot*, a partir de un patrón de entrada, lo que le permitirá a la segunda etapa recuperar de manera acertada el patrón correspondiente. El proceso en el sentido hacia atrás es similar al descrito previamente.

La complejidad mostrada por el algoritmo utilizado para implementar la Memoria Asociativa Bidireccional Alfa-Beta es $O(n^2)$.

Se presentan tres ejemplos de aplicación. El primero permite elegir parejas de patrones de entre 4 conjuntos de 26 patrones cada uno. El segundo ejemplo es el reconocimiento de huellas dactilares que se asocian a números; en este caso de utilizan 40 parejas de patrones. La tercera aplicación es un traductor inglés-español/español-inglés que trabaja sobre 120 parejas de palabras. En las tres aplicaciones, y como es de esperarse, el modelo presentó recuperación correcta.

# Abstract

In this work, a new model for Bidirectional Associative Memories (BAM), called Alpha-Beta BAM, is presented. The model is based on the Alpha-Beta Associative Memories, from which it inherits its name.

Due to its origin, Alpha-Beta BAM do not require an iterative process of convergence toward stable states, unlike the models based on Kosko's BAM. Because of this, they have no stability problems.

The patterns used by the Alpha-Beta BAM are binary patterns.

Recall capacity of the proposed model is $2^{\min(n,m)}$, being $n$ and $m$ the respective dimensions of input and output patterns. Therefore, the Alpha-Beta BAM always recover correctly all trained patterns. Proper recall requires no previous condition on the properties of patterns, such as Hamming distance, orthogonality, or linear separation.

Mathematical foundation which supports why Alpha-Beta BAM always have correct recall is presented.

Two new vector transformations are proposed: *expansion* and *contraction*.

During the learning phase, Alpha-Beta BAM make use of the *one-hot* binary code and a proposed new code: the *zero-hot* binary code.

The Alpha-Beta BAM are made up of four stages; two stages are used in the forward direction, and the other two stages are used in the backward direction, thus allowing the biderectionality of the model.

The first stage, in the forward direction, is composed of two autoassociative Alpha-Beta memories, *max* and *min*, built from the input patterns. On the second stage a modified *Linear Associator* is built from the output patterns. The task of the first stage is to obtain a *one-hot* vector, from an input pattern, which in turn allows the second stage to correctly recall the corresponding pattern. The process in the backward direction is similar to the one described above.

The complexity shown by the algorithm used to implement the Alpha-Beta Bidirectional Associative memory is $O(n^2)$.

Three examples of application are presented. The first one allows to choose pairs of patterns between 4 sets of 26 patterns each. The second example is the recognition of fingerprints associated to numbers; in this case 40 pairs of patterns are used. The third applications is a englis-spanish/spanish-english translator which work with 120 pairs of words. In the three applications, as was expected, the model showed proper recall.

# CAPÍTULO 1

# Introducción

## 1.1 Contexto

El área de las Memorias Asociativas, como parte relevante de las Ciencias de la Computación, ha adquirido gran importancia y dinamismo en la actividad desarrollada por numerosos equipos de investigación internacionales, específicamente en los que investigan temas relacionados con la teoría y aplicaciones del reconocimiento y clasificación de patrones.

El propósito fundamental de una memoria asociativa es recuperar correctamente patrones completos a partir de patrones de entrada, los cuales pueden estar alterados con ruido aditivo, sustractivo o combinado [1]. En el diseño de una memoria asociativa, previamente a la fase de recuperación de patrones, se requiere la fase de aprendizaje, que es el proceso mediante el cual se crea la memoria asociativa a través de la formación de asociaciones de patrones, las cuales son parejas de patrones, uno de entrada y uno de salida. Si en cada asociación sucede que el patrón de entrada es igual al de salida, la memoria es autoasociativa; en caso contrario, la memoria es heteroasociativa: esto significa que una memoria autoasociativa puede considerarse como un caso particular de una memoria heteroasociativa [2].

A través del tiempo las Memorias Asociativas se han desarrollado paralelamente a las Redes Neuronales, desde la concepción del primer modelo de neurona artificial [3], hasta los modelos de redes neuronales basados en conceptos modernos como la Morfología Matemática [4], pasando por los importantes trabajos de los pioneros en las redes neuronales tipo *perceptron* [5, 6, 7].

En 1982 John J. Hopfield presenta al mundo su memoria asociativa, modelo que fue inspirado en conceptos físicos y tiene como particularidad un algoritmo iterativo [8]. Este trabajo propició el renacimiento del interés de los investigadores en los temas tanto de memorias asociativas como de redes neuronales, los cuales se habían dejado de lado por algunos años. Las redes neuronales tuvieron un receso de 13 años, a partir de que Minsky y Papert, en 1969, publicaron su libro *Perceptrons* [9] en donde demostraron que el *perceptron* tenía severas limitaciones. Por el lado de las memorias asociativas, fue Karl Steinbuch quien por primera vez, en 1961, desarrolla una memoria heteroasociativa que funciona como un clasificador de patrones binarios: la *Lernmatrix* [10]. Ocho años después, los investigadores escoceses Willshaw, Buneman y Longuet-Higgins [11] presentan el *Correlograph*, dispositivo óptico elemental capaz de funcionar como una memoria asociativa. Existe otro antecedente importante de la memoria Hopfield: dos modelos

clásicos de memorias asociativas fueron presentados en 1972 por Anderson [12] y Kohonen [13] de manera independiente, y debido a su importancia y a la similitud de los conceptos involucrados, ambos modelos reciben el nombre genérico de *Linear Associator*.

El trabajo de Hopfield tiene gran relevancia debido a que el modelo de red neuronal que presenta, demuestra que la interacción de elementos simples de procesamiento similares a las neuronas dan lugar a la aparición de propiedades computacionales colectivas, tales como la estabilidad de memorias.

Sin embargo, el modelo Hopfield de memoria asociativa tiene dos desventajas: primeramente es notable el hecho de que la capacidad de recuperación de patrones es muy pequeña, sólo de $0.15n$ siendo $n$ la dimensión de los patrones almacenados; y en segundo lugar, la memoria Hopfield es sólo autoasociativa; es decir, no es capaz de asociar patrones diferentes.

Con objeto de subsanar la segunda desventaja del modelo Hopfield, en 1988 Bart Kosko [14] crea un modelo de memoria heteroasociativa a partir de la memoria Hopfield: la memoria asociativa bidireccional BAM (Bidirectional Associative Memory) la cual, al igual que la memoria Hopfield, se basa en un algoritmo iterativo. Este modelo teórico es la base que motivó el tema central de este trabajo de tesis.

## 1.2 El problema a resolver

La capacidad de aprendizaje y almacenamiento, la eficiencia en la respuesta o recuperación de patrones, la rapidez y la inmunidad al ruido, son tópicos de interés entre los investigadores que se dedican a estudiar los modelos de memoria asociativa con el fin de proponer variaciones y generalizaciones que, a la postre, se traduzcan en nuevos modelos con ventajas claras sobre los ya conocidos y que, además, sean propicios para su aplicación directa en problemas reales [15].

En el caso de la BAM de Kosko, la característica fundamental es la heteroasociatividad que se logró con base en el modelo de Hopfield. Kosko propuso una solución basada en la propia matriz que representaba la memoria Hopfield, con la cual fue capaz de realizar la fase de aprendizaje en ambas direcciones: obtener un patrón de salida a partir del correspondiente patrón de entrada o, viceversa, obtener un patrón de entrada a partir del correspondiente patrón de salida; de ahí el nombre de bidireccional.

No obstante que el intento de Kosko fue exitoso al obtener una memoria heteroasociativa, la otra desventaja ya mencionada de la memoria Hopfield no fue subsanada con la BAM: la memoria asociativa bidireccional de Kosko exhibe una muy baja capacidad de aprendizaje y recuperación de patrones, dependiente del mínimo de las dimensiones de los patrones de entrada y salida.

En el mismo año de la aparición de la BAM, 1988, surgieron los primeros intentos de investigación científica para mejorar la capacidad de aprendizaje y recuperación de patrones de la BAM [16]. A partir de entonces diversos grupos de investigación de muchas

partes del mundo han intentado, a través de los años, minimizar esa desventaja de la BAM, mediante la utilización de los más variados conceptos científicos y técnicas matemáticas, desde el entrenamiento múltiple hasta codificación Householder [17] y algoritmos genéticos [18], y en algunos casos programación lineal.

Sin embargo, los avances han sido modestos y escasos, al grado de que la gran mayoría de las propuestas de nuevos modelos de memorias asociativas bidireccionales planteadas en revistas científicas y congresos de alto nivel, no garantizan ni siquiera la recuperación completa del conjunto fundamental, de aprendizaje o de entrenamiento, utilizado en la fase de aprendizaje: es decir, estos modelos no son capaces de recuperar todos los patrones aprendidos, y en uno o más fallan [14],[16-19], [28-65], [76-84].

La investigación científica que sustenta este trabajo de tesis es una respuesta a lo anterior. El problema a resolver en este trabajo de tesis es: encontrar un modelo de memoria asociativa bidireccional que exhiba eficacia en el aprendizaje y recuperación de patrones; un modelo que recupere, al menos, todos los patrones del conjunto fundamental de aprendizaje, tanto de entrada como de salida, y de preferencia que se base en un algoritmo *one-shot*, es decir, un algoritmo no iterativo, que no requiera convergencia.

## 1.3 Objetivo

Diseñar e implementar un nuevo modelo teórico de memoria asociativa bidireccional, que exhiba recuperación perfecta de todos los patrones del conjunto fundamental de aprendizaje, con gran capacidad de almacenamiento y que se base en un algoritmo *one shot*; es decir, un algoritmo no iterativo, que no requiera convergencia.

## 1.4 Contribuciones

La aportación principal de este trabajo de tesis al bagaje científico en el área de las ciencias de la computación, es un nuevo modelo de memoria asociativa bidireccional que exhibe recuperación perfecta de todos los patrones del conjunto fundamental de aprendizaje, y que además se basa en un algoritmo no iterativo, que no requiere convergencia.

Una contribución adicional es una modificación original al algoritmo del *Linear Associator* cuando el conjunto fundamental está formado por vectores *one-hot*.

Paralelamente, se plantean algunas ideas de aplicación de este nuevo modelo en áreas específicas de la investigación científica y tecnológica.

## 1.5 Organización del documento

En este capítulo se han presentado: el contexto, el problema a resolver, el objetivo del trabajo de tesis y las contribuciones. El resto del documento de tesis está organizado como sigue:

En el capítulo 2 se presentan los conceptos básicos de las memorias asociativas, así como los modelos más representativos que sirven como marco de referencia de los modelos de memorias asociativas bidireccionales. Además, se presenta el estado del arte de los modelos de memorias asociativas bidireccionales más importantes a través del tiempo.

En el capítulo 3 se describen, en detalle, las Memorias Asociativas Alfa-Beta, modelo que representa el pilar teórico y conceptual fundamental en el diseño y operación de las Memorias Asociativas Bidireccionales Alfa-Beta.

En el capítulo 4 se describe el nuevo modelo original: Memorias Asociativas Bidireccionales Alfa-Beta, tema central de este trabajo de tesis. Además, se da el fundamento matemático original que sustenta el diseño y operación de las cuatro etapas que constituyen el modelo. Posteriormente, se detalla el algoritmo que permite el aprendizaje y recuperación de patrones de la Memoria Asociativa Bidireccional Alfa-Beta. Por último, se incluye un análisis de complejidad del nuevo modelo propuesto.

En el capítulo 5 se presentan aspectos experimentales: primeramente, se comparan los resultados obtenidos por dos modelos de BAM cuyo algoritmo se basa en el modelo de Kosko, y los resultados que arroja la BAM Alfa-Beta al aplicarse a un ejemplo numérico bastante ilustrativo. Acto seguido, se incluye otra comparación, del tiempo de ejecución, con una BAM que utiliza algoritmos genéticos en el diseño de su modelo. Además, se muestran tres aplicaciones de la Memoria Asociativa Alfa-Beta: la primera aplicación permite crear conjuntos fundamentales de patrones monocromáticos, los cuales son entrenados y recuperados de manera correcta por la BAM Alfa-Beta; la segunda aplicación es un identificador de huellas digitales, y la tercera es un traductor inglés-español/español-inglés. Como colofón, se hace hincapié en la característica más importante del nuevo modelo: la recuperación correcta de todos los patrones del conjunto fundamental, y se presentan gráficas comparativos con otros modelos representativos de BAM.

En el capítulo 6, se presentan las conclusiones finales obtenidas a partir del desarrollo y operación del nuevo modelo original de Memoria Asociativa Bidireccional Alfa-Beta; se incluyen, además, algunos posibles temas para trabajos de investigación derivados del nuevo modelo propuesto en este trabajo de tesis. Posteriormente, se presenta la relación de las publicaciones propias derivadas del nuevo modelo propuesto.

Finalmente, se presentan dos apéndices: el apéndice A con la simbología usada en el escrito, y el apéndice B, que contiene la descripción detallada del estado del arte de las Memorias Asociativas Bidireccionales; por último, se incluyen las referencias bibliográficas.

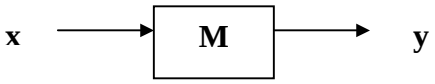
# CAPÍTULO 2

# Antecedentes

## 2.1 Conceptos Básicos

El propósito fundamental de una memoria asociativa es recuperar correctamente patrones completos a partir de patrones de entrada, los cuales pueden estar alterados con ruido aditivo, sustractivo o combinado. Los conceptos utilizados en esta sección se encuentran en las referencias [1, 2, 15].

Una **Memoria Asociativa** puede formularse como un sistema de entrada y salida, idea que se esquematiza a continuación:




En este esquema, los patrones de entrada y salida están representados por vectores columna denotados por **x** y **y**, respectivamente. Cada uno de los patrones de entrada forma una asociación con el correspondiente patrón de salida, la cual es similar a la una pareja ordenada; por ejemplo, los patrones **x** y **y** del esquema anterior forman la asociación (**x**,**y**).

No obstante que a lo largo de las dos secciones restantes del presente capítulo se respetarán las notaciones originales de los autores de los modelos presentados aquí, a continuación se propone una notación que se usará en la descripción de los conceptos básicos sobre memorias asociativas, y en el resto de los capítulos de esta tesis.

Los patrones de entrada y salida se denotarán con las letras negrillas, **x** y **y**, agregándoles números naturales como superíndices para efectos de discriminación simbólica. Por ejemplo, a un patrón de entrada $\mathbf{x}^1$ le corresponderá el patrón de salida $\mathbf{y}^1$, y ambos formarán la asociación $(\mathbf{x}^1,\mathbf{y}^1)$; del mismo modo, para un número entero positivo $k$ específico, la asociación correspondiente será $(\mathbf{x}^k,\mathbf{y}^k)$.

La memoria asociativa **M** se representa mediante una matriz, la cual se genera a partir de un conjunto finito de asociaciones conocidas de antemano: este es el **conjunto fundamental de aprendizaje**, o simplemente **conjunto fundamental**.

El conjunto fundamental se representa de la siguiente manera:

$$\{(\mathbf{x}^\mu, \mathbf{y}^\mu) \mid \mu = 1, 2, ..., p\}$$

donde $p$ es un número entero positivo que representa la cardinalidad del conjunto fundamental.

A los patrones que conforman las asociaciones del conjunto fundamental se les llama **patrones fundamentales**. La naturaleza del conjunto fundamental proporciona un importante criterio para clasificar las memorias asociativas:

Una memoria es **Autoasociativa** si se cumple que $\mathbf{x}^{\mu} = \mathbf{y}^{\mu} \ \forall \mu \in \{1, 2, ..., p\}$, por lo que uno de los requisitos que se debe de cumplir es que $n = m$.

Una memoria **Heteroasociativa** es aquella en donde $\exists \mu \in \{1, 2, ..., p\}$ para el que se cumple que $\mathbf{x}^{\mu} \neq \mathbf{y}^{\mu}$. Nótese que puede haber memorias heteroasociativas con $n = m$.

En los problemas donde intervienen las memorias asociativas, se consideran dos fases importantes: La fase de aprendizaje, que es donde se genera la memoria asociativa a partir de las $p$ asociaciones del conjunto fundamental, y la fase de recuperación que es donde la memoria asociativa opera sobre un patrón de entrada, a la manera del esquema que aparece al inicio de esta sección.

A fin de especificar las componentes de los patrones, se requiere la notación para dos conjuntos a los que llamaremos arbitrariamente $A$ y $B$. Las componentes de los vectores columna que representan a los patrones, tanto de entrada como de salida, serán elementos del conjunto $A$, y las entradas de la matriz $\mathbf{M}$ serán elementos del conjunto $B$.

No hay requisitos previos ni limitaciones respecto de la elección de estos dos conjuntos, por lo que no necesariamente deben ser diferentes o poseer características especiales. Esto significa que el número de posibilidades para escoger $A$ y $B$ es infinito.

Por convención, cada vector columna que representa a un patrón de entrada tendrá $n$ componentes cuyos valores pertenecen al conjunto $A$, y cada vector columna que representa a un patrón de salida tendrá $m$ componentes cuyos valores pertenecen también al conjunto $A$. Es decir:

$$\mathbf{x}^{\mu} \in A^n \text{ y } \mathbf{y}^{\mu} \in A^m \ \forall \mu \in \{1, 2, ..., p\}$$

La $j$-ésima componente de un vector columna se indicará con la misma letra del vector, pero sin negrilla, colocando a $j$ como subíndice ($j \in \{1, 2, ..., n\}$ o $j \in \{1, 2, ..., m\}$ según corresponda). La $j$-ésima componente del vector columna $\mathbf{x}^{\mu}$ se representa por: $x_j^{\mu}$

Con los conceptos básicos ya descritos y con la notación anterior, es posible expresar las dos fases de una memoria asociativa:

1. **Fase de Aprendizaje** (Generación de la memoria asociativa). Encontrar los operadores adecuados y una manera de generar una matriz $\mathbf{M}$ que almacene las $p$ asociaciones del conjunto fundamental $\{(\mathbf{x}^1,\mathbf{y}^1), (\mathbf{x}^2,\mathbf{y}^2), ..., (\mathbf{x}^p,\mathbf{y}^p)\}$, donde $\mathbf{x}^{\mu} \in A^n$ y $\mathbf{y}^{\mu} \in A^m \ \forall \mu \in \{1, 2, ..., p\}$. Si $\exists \mu \in \{1, 2, ..., p\}$ tal que $\mathbf{x}^{\mu} \neq \mathbf{y}^{\mu}$, la memoria será

heteroasociativa; si $m = n$ y $\mathbf{x}^{\mu} = \mathbf{y}^{\mu}$ $\forall \mu \in \{1, 2, ..., p\}$, la memoria será autoasociativa.

2. **Fase de Recuperación** (Operación de la memoria asociativa). Hallar los operadores adecuados y las condiciones suficientes para obtener el patrón fundamental de salida $\mathbf{y}^{\mu}$, cuando se opera la memoria **M** con el patrón fundamental de entrada $\mathbf{x}^{\mu}$; lo anterior para todos los elementos del conjunto fundamental y para ambos modos: autoasociativo y heteroasociativo.

Se dice que una memoria asociativa **M** exhibe **recuperación correcta** si al presentarle como entrada, en la fase de recuperación, un patrón $\mathbf{x}^{\omega}$ con $\omega \in \{1, 2, ..., p\}$, ésta responde con el correspondiente patrón fundamental de salida $\mathbf{y}^{\omega}$.

Una memoria asociativa bidireccional también es un sistema de entrada y salida, solamente que el proceso es bidireccional. La dirección hacia adelante se describe de la misma forma que una memoria asociativa común: al presentarle una entrada **x**, el sistema entrega una salida **y**. La dirección hacia atrás se lleva a cabo presentándole al sistema una entrada **y** para recibir una salida **x**.

## 2.2 Memorias Asociativas

A continuación, en esta sección haremos un breve recorrido por los modelos de memorias asociativas, con objeto de establecer el marco de referencia en el que surgieron las memorias asociativas bidireccionales.

Las memorias asociativas que se presentarán en esta sección, son los modelos más representativos que sirvieron de base para la creación de modelos matemáticos que sustentan el diseño y operación de memorias asociativas más complejas. Para cada modelo se describe su fase de aprendizaje y su fase de recuperación.

Se incluyen cuatro modelos clásicos basados en el anillo de los números racionales con las operaciones de multiplicación y adición: *Lernmatrix*, *Correlograph*, *Linear Associator* y Memoria Hopfield, además de tres modelos basados en paradigmas diferentes a la suma de productos, a saber: memorias asociativas Morfológicas, memorias asociativas Alfa-Beta y memorias asociativas Media.

### 2.2.1 *Lernmatrix* de Steinbuch

Karl Steinbuch fue uno de los primeros investigadores en desarrollar un método para codificar información en arreglos cuadriculados conocidos como *crossbar* [20]. La importancia de la *Lernmatrix* [9, 21] se evidencia en una afirmación que hace Kohonen [13] en su artículo de 1972, donde apunta que las matrices de correlación, base fundamental de su innovador trabajo, vinieron a sustituir a la *Lernmatrix* de Steinbuch.

La *Lernmatrix* es una memoria heteroasociativa que puede funcionar como un clasificador de patrones binarios si se escogen adecuadamente los patrones de salida; es un sistema de

entrada y salida que al operar acepta como entrada un patrón binario $\mathbf{x}^{\mu} \in A^n, \quad A = \{0,1\}$ y produce como salida la clase $\mathbf{y}^{\mu} \in A^p$ que le corresponde (de entre *p* clases diferentes), codificada ésta con un método que en la literatura se le ha llamado *one-hot* [22]. El método funciona así: para representar la clase $k \in \{1, 2, ..., p\}$, se asignan a las componentes del vector de salida $\mathbf{y}^{\mu}$ los siguientes valores: $y_k^{\mu} = 1$, y $y_j^{\mu} = 0$ para $j = 1, 2, ..., k\text{-}1, k\text{+}1, ..., p$.

### Algoritmo de la *Lernmatrix*

#### Fase de Aprendizaje

Se genera el esquema (*crossbar*) al incorporar la pareja de patrones de entrenamiento $(\mathbf{x}^{\mu}, \mathbf{y}^{\mu}) \in A^n \times A^p$. Cada uno de los componentes $m_{ij}$ de **M**, la *Lernmatrix* de Steinbuch, tiene valor cero al inicio, y se actualiza de acuerdo con la regla $m_{ij} + \Delta m_{ij}$, donde:

$$\Delta m_{ij} = \begin{cases} +\varepsilon & \text{si } x_j^{\mu} = 1 = y_i^{\mu} \\ -\varepsilon & \text{si } x_j^{\mu} = 0 \text{ y } y_i^{\mu} = 1 \\ 0 & \text{en otro caso} \end{cases}$$

donde $\varepsilon$ una constante positiva escogida previamente: es usual que $\varepsilon$ es igual a 1.

#### Fase de Recuperación

La *i*-ésima coordenada $y_i^{\omega}$ del vector de clase $\mathbf{y}^{\omega} \in A^p$ se obtiene como lo indica la siguiente expresión, donde $\vee$ es el operador *máximo*:

$$y_i^{\omega} = \begin{cases} 1 & \text{si } \sum_{j=1}^{n} m_{ij} . x_j^{\omega} = \bigvee_{h=1}^{p} \left[ \sum_{j=1}^{n} m_{hj} . x_j^{\omega} \right] \\ 0 & \text{en otro caso} \end{cases}$$

### 2.2.2 *Correlograph* de Willshaw, Buneman y Longuet-Higgins

El *correlograph* es un dispositivo óptico elemental capaz de funcionar como una memoria asociativa [11]. En palabras de los autores “el sistema es tan simple, que podría ser construido en cualquier laboratorio escolar de física elemental”.

### Algoritmo del *Correlograph*

#### Fase de Aprendizaje

La *red asociativa* se genera al incorporar la pareja de patrones de entrenamiento $(\mathbf{x}^{\mu}, \mathbf{y}^{\mu}) \in A^n \times A^m$. Cada uno de los componentes $m_{ij}$ de la *red asociativa* **M** tiene valor cero al inicio, y se actualiza de acuerdo con la regla:

$$m_{ij} = \begin{cases} 1 & \text{si } y_i^\mu = 1 = x_j^\mu \\ \textit{valor anterior} & \text{en otro caso} \end{cases}$$

### Fase de Recuperación

Se le presenta a la *red asociativa* **M** un vector de entrada $\mathbf{x}^\omega \in A^n$. Se realiza el producto de la matriz **M** por el vector $\mathbf{x}^\omega$ y se ejecuta una operación de umbralizado, de acuerdo con la siguiente expresión:

$$y_i^\omega = \begin{cases} 1 & \text{si } \sum_{j=1}^n m_{ij}.x_j^\omega \geq u \\ 0 & \text{en otro caso} \end{cases}$$

donde *u* es el valor de umbral. Una estimación aproximada del valor de umbral *u* se puede lograr con la ayuda de un número indicador mencionado en el artículo [11] de Willshaw *et al*. de 1969: $log_2 n$

### 2.2.3 *Linear Associator* de Anderson-Kohonen

El *Linear Associator* tiene su origen en los trabajos pioneros de 1972 publicados por Anderson y Kohonen [12, 13].

Para presentar el *Linear Associator* consideremos de nuevo el conjunto fundamental:

$$\{(\mathbf{x}^\mu, \mathbf{y}^\mu) \mid \mu = 1, 2, ..., p\} \text{ con } A = \{0, 1\}, \quad \mathbf{x}^\mu \in A^n \quad \text{y} \quad \mathbf{y}^\mu \in A^m$$

#### Algoritmo del *Linear Associator*

### Fase de Aprendizaje

1) Para cada una de las *p* asociaciones $(\mathbf{x}^\mu, \mathbf{y}^\mu)$ se encuentra la matriz $\mathbf{y}^\mu \cdot \left(\mathbf{x}^\mu\right)^t$ de dimensiones *m* x *n*

2) Se suman la *p* matrices para obtener la memoria

$$\mathbf{M} = \sum_{\mu=1}^{p} \mathbf{y}^\mu \cdot \left(\mathbf{x}^\mu\right)^t = \left[m_{ij}\right]_{m \times n}$$

de manera que la *ij*-ésima componente de la memoria M se expresa así:

$$m_{ij} = \sum_{\mu=1}^{p} y_i^\mu x_j^\mu$$

### Fase de Recuperación

Esta fase consiste en presentarle a la memoria un patrón de entrada $\mathbf{x}^{\omega}$, donde $\omega \in \{1, 2, ..., p\}$ y realizar la operación

$$\mathbf{M} \cdot \mathbf{x}^{\omega} = \left[ \sum_{\mu=1}^{p} \mathbf{y}^{\mu} \cdot \left( \mathbf{x}^{\mu} \right)^{t} \right] \cdot \mathbf{x}^{\omega}$$

### 2.2.4 La memoria asociativa Hopfield

El artículo de John J. Hopfield de 1982, publicado por la prestigiosa y respetada *National Academy of Sciences* (en sus *Proceedings*), impactó positivamente y trajo a la palestra internacional su famosa memoria asociativa [8].

En el modelo que originalmente propuso Hopfield, cada neurona $x_i$ tiene dos posibles estados, a la manera de las neuronas de McCulloch-Pitts: $x_i = 0$ y $x_i = 1$; sin embargo, Hopfield observa que, para un nivel dado de exactitud en la recuperación de patrones, la capacidad de almacenamiento de información de la memoria se puede incrementar por un factor de 2, si se escogen como posibles estados de las neuronas los valores $x_i = -1$ y $x_i = 1$ en lugar de los valores originales $x_i = 0$ y $x_i = 1$.

Al utilizar el conjunto {-1, 1} y el valor de umbral cero, la fase de aprendizaje para la memoria Hopfield será similar, en cierta forma, a la fase de aprendizaje del *Linear Associator*. La intensidad de la fuerza de conexión de la neurona $x_i$ a la neurona $x_j$ se representa por el valor de $m_{ij}$, y se considera que hay simetría, es decir, $m_{ij} = m_{ji}$. Si $x_i$ no está conectada con $x_j$ entonces $m_{ij} = 0$; en particular, no hay conexiones recurrentes de una neurona a sí misma, lo cual significa que $m_{ij} = 0$. El estado instantáneo del sistema está completamente especificado por el vector columna de dimensión $n$ cuyas coordenadas son los valores de las $n$ neuronas.

La memoria Hopfield es autoasociativa, simétrica, con ceros en la diagonal principal. En virtud de que la memoria es autoasociativa, el conjunto fundamental para la memoria Hopfield es $\{(\mathbf{x}^{\mu}, \mathbf{x}^{\mu}) \mid \mu = 1, 2, ..., p\}$ con $\mathbf{x}^{\mu} \in A^n$ y $A = \{-1, 1\}$

#### Algoritmo Hopfield

### Fase de Aprendizaje

La fase de aprendizaje para la memoria Hopfield es similar a la fase de aprendizaje del *Linear Associator*, con una ligera diferencia relacionada con la diagonal principal en ceros, como se muestra en la siguiente regla para obtener la $ij$-ésima componente de la memoria Hopfield **M**:

$$m_{ij} = \begin{cases} \sum_{\mu=1}^{p} x_i^{\mu} x_j^{\mu} \text{ si } i \neq j \\ 0 \text{ si } i = j \end{cases}$$

### Fase de Recuperación

Si se le presenta un patrón de entrada $\tilde{\mathbf{x}}$ a la memoria Hopfield, ésta cambiará su estado con el tiempo, de modo que cada neurona $x_i$ ajuste su valor de acuerdo con el resultado que arroje la comparación de la cantidad $\sum_{j=1}^{n} m_{ij} x_j$ con un valor de umbral, el cual normalmente se coloca en cero.

Se representa el estado de la memoria Hopfield en el tiempo $t$ por $\mathbf{x}(t)$; entonces $x_i(t)$ representa el valor de la neurona $x_i$ en el tiempo $t$ y $x_i(t+1)$ el valor de $x_i$ en el tiempo siguiente $(t+1)$.

Dado un vector columna de entrada $\tilde{\mathbf{x}}$, la fase de recuperación consta de tres pasos:

1) Para $t = 0$, se hace $\mathbf{x}(t) = \tilde{\mathbf{x}}$; es decir, $x_i(0) = \tilde{x}_i$, $\forall i \in \{1,2,3,...,n\}$
2) $\forall i \in \{1,2,3,...,n\}$ se calcula $x_i(t+1)$ de acuerdo con la condición siguiente:

$$x_i(t+1) = \begin{cases} 1 \text{ si } \sum_{j=1}^{n} m_{ij} x_j(t) > 0 \\ x_i(t) \text{ si } \sum_{j=1}^{n} m_{ij} x_j(t) = 0 \\ -1 \text{ si } \sum_{j=1}^{n} m_{ij} x_j(t) < 0 \end{cases}$$

3) Se compara $x_i(t+1)$ con $x_i(t)$ $\forall i \in \{1, 2, 3,...,n\}$. Si $\mathbf{x}(t+1) = \mathbf{x}(t)$ el proceso termina y el vector recuperado es $\mathbf{x}(0) = \tilde{\mathbf{x}}$. De otro modo, el proceso continúa de la siguiente manera: los pasos 2 y 3 se iteran tantas veces como sea necesario hasta llegar a un valor $t = \tau$ para el cual $x_i(\tau+1) = x_i(\tau)$ $\forall i \in \{1, 2, 3,...,n\}$; el proceso termina y el patrón recuperado es $\mathbf{x}(\tau)$.

En el artículo original de 1982, Hopfield había estimado empíricamente que su memoria tenía una capacidad de recuperar $0.15n$ patrones, y en el trabajo de Abu-Mostafa & St. Jacques [24] se estableció formalmente que una cota superior para el número de vectores de estado arbitrarios estables en una memoria Hopfield es $n$.

### 2.2.5 Memorias Asociativas Morfológicas

La diferencia fundamental entre las memorias asociativas clásicas (*Lernmatrix*, *Correlograph*, *Linear Associator* y Memoria Asociativa Hopfield) y las memorias asociativas morfológicas radica en los fundamentos operacionales de éstas últimas, que son las operaciones morfológicas de *dilatación* y *erosión*; el nombre de las memorias asociativas morfológicas está inspirado precisamente en estas dos operaciones básicas. Estas memorias rompieron con el esquema utilizado a través de los años en los modelos de memorias asociativas clásicas, que utilizan operaciones convencionales entre vectores y matrices para la fase de aprendizaje y suma de productos para la recuperación de patrones. Las memorias asociativas morfológicas cambian los productos por sumas y las sumas por máximos o mínimos en ambas fases, tanto de aprendizaje como de recuperación [25].

Hay dos tipos de memorias asociativas morfológicas: las memorias *max*, simbolizadas con **M**, y las memorias *min*, cuyo símbolo es **W**; en cada uno de los dos tipos, las memorias pueden funcionar en ambos modos: heteroasociativo y autoasociativo.

Se definen dos nuevos productos matriciales:
El *producto máximo* entre **D** y **H**, denotado por $\mathbf{C} = \mathbf{D} \nabla \mathbf{H}$, es una matriz $[c_{ij}]_{mxn}$ cuya *ij*-ésima componente $c_{ij}$ es

$$c_{ij} = \bigvee_{k=1}^{r} \left( d_{ik} + h_{kj} \right)$$

El *producto mínimo* de **D** y **H** denotado por $\mathbf{C} = \mathbf{D} \Delta \mathbf{H}$, es una matriz $[c_{ij}]_{mxn}$ cuya *ij*-ésima componente $c_{ij}$ es

$$c_{ij} = \bigwedge_{k=1}^{r} \left( d_{ik} + h_{kj} \right)$$

Los productos máximo y mínimo contienen a los operadores máximo y mínimo, los cuales están íntimamente ligados con los conceptos de las dos operaciones básicas de la morfología matemática: *dilatación* y *erosión*, respectivamente.

#### 2.2.5.1 Memorias Heteroasociativas Morfológicas

**Algoritmo de las memorias morfológicas *max***

**Fase de Aprendizaje**

1. Para cada una de las *p* asociaciones $(\mathbf{x}^{\mu}, \mathbf{y}^{\mu})$ se usa el producto mínimo para crear la matriz $\mathbf{y}^{\mu} \Delta (-\mathbf{x}^{\mu})^{t}$ de dimensiones *m* x *n*, donde el negado transpuesto del patrón de entrada $\mathbf{x}^{\mu}$ se define como $(-\mathbf{x}^{\mu})^{t} = \left( -x_1^{\mu}, -x_2^{\mu}, ..., x_n^{\mu} \right)$.
2. Se aplica el operador máximo $\vee$ a las *p* matrices para obtener la memoria **M**.

$$\mathbf{M} = \bigvee_{\mu=1}^{p} \left[ \mathbf{y}^{\mu} \Delta (-\mathbf{x}^{\mu})^{t} \right]$$

### Fase de Recuperación

Esta fase consiste en realizar el producto mínimo $\Delta$ de la memoria **M** con el patrón de entrada $\mathbf{x}^{\omega}$, donde $\omega \in \{1, 2, ..., p\}$, para obtener un vector columna **y** de dimensión *m*:

$$\mathbf{y} = \mathbf{M} \Delta \mathbf{x}^{\omega}$$

Las fases de aprendizaje y de recuperación de las **memorias morfológicas *min*** se obtienen por dualidad.

#### 2.2.5.2 Memorias Autoasociativas Morfológicas

Para este tipo de memorias se utilizan los mismos algoritmos descritos anteriormente y que son aplicados a las memorias heteroasociativas; lo único que cambia es el conjunto fundamental. Para este caso, se considera el siguiente conjunto fundamental:

$$\{(\mathbf{x}^{\mu},\mathbf{x}^{\mu}) \mid \mathbf{x}^{\mu} \in A^{n}, \text{ donde } \mu = 1, 2, ..., p\}$$

#### 2.2.6 Memorias Asociativas Alfa-Beta

Las memorias Alfa-Beta [15] utilizan máximos y mínimos, y dos operaciones binarias originales $\alpha$ y $\beta$ de las cuales heredan el nombre.

Para la definición de las operaciones binarias $\alpha$ y $\beta$ se deben especificar los conjuntos *A* y *B*, los cuales son:

$$A = \{0, 1\} \quad \text{y} \quad B = \{0, 1, 2\}$$

La operación binaria $\alpha$: $A \times A \rightarrow B$ se define como:

| *x* | *y* | $\alpha(x, y)$ |
|---|---|---|
| 0 | 0 | 1 |
| 0 | 1 | 0 |
| 1 | 0 | 2 |
| 1 | 1 | 1 |

La operación binaria $\beta$: $B \times A \rightarrow A$ se define como:

| *x* | *y* | $\beta(x, y)$ |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 1 |
| 2 | 0 | 1 |
| 2 | 1 | 1 |

Los conjuntos *A* y *B*, las operaciones binarias $\alpha$ y $\beta$ junto con los operadores $\wedge$ (mínimo) y $\vee$ (máximo) usuales conforman el sistema algebraico (*A*, *B*, $\alpha$, $\beta$, $\wedge$, $\vee$) en el que están inmersas las memorias asociativas Alfa-Beta.

El fundamento teórico de las memorias asociativas Alfa-Beta se presenta en el siguiente capítulo de forma más completa, debido a que estas memorias son la base fundamental para la construcción del modelo de BAM propuesto en este trabajo de tesis.

### 2.2.7 Memorias Asociativas Media

Las Memorias Asociativas Media [26] utilizan los operadores A y B, definidos de la siguiente forma:

$$A(x, y) = x - y$$
$$B(x, y) = x + y$$

Las operaciones utilizadas se describen a continuación.

Sean $P = [p_{ij}]_{mxr}$ y $Q=[q_{ij}]_{rxn}$ dos matrices.

Operación $\lozenge_A$: $P_{mxr} \lozenge_A Q_{rxn} = [f_{ij}^A]_{mxn}$ donde $f_{ij}^A = \underset{k=1}{\overset{r}{\mathbf{med}}} A(p_{ik}, q_{k,j})$

Operación $\lozenge_B$: $P_{mxr} \lozenge_B Q_{rxn} = [f_{ij}^B]_{mxn}$ donde $f_{ij}^B = \underset{k=1}{\overset{r}{\mathbf{med}}} B(p_{ik}, q_{k,j})$

**Algoritmo Memorias Media**

**Fase de Aprendizaje**

**Paso 1.** Para cada $\xi = 1, 2, ..., p$, de cada pareja $(\mathbf{x}^\xi, \mathbf{y}^\xi)$ se construye la matriz:

$$[\mathbf{y}^\xi \lozenge_A (\mathbf{x}^\xi)^t]_{mxn}$$

**Paso 2.** Se aplica el operador media a las matrices obtenidas en le paso 1 para obtener la matriz **M**, como sigue:

$$\mathbf{M} = \underset{\xi=1}{\overset{p}{\mathbf{med}}} \left[ y^\xi \lozenge_A \left( x^\xi \right)^t \right]$$

El *ij*-ésimo componente **M** está dado como sigue:

$$m_{ij} = \underset{\xi=1}{\overset{p}{\mathbf{med}}} A\left( y_i^\xi, x_j^\xi \right)$$

### Fase de Recuperación

Se tienen dos casos:

**Caso 1.** Recuperación de un patrón fundamental. Un patrón $\mathbf{x}^w$, con $w \in \{1, 2, ..., p\}$ se le presenta a la memoria $\mathbf{M}$ y se realiza la siguiente operación:

$$\mathbf{M} \lozenge_{\mathrm{B}} \mathbf{x}^w$$

El resultado es un vector columna de dimensión $n$, con la $i$-ésima componente dada como:

$$\left(\mathbf{M} \lozenge_B x^w\right)_i = \underset{j=1}{\overset{n}{\mathbf{med}}} \mathrm{B}\left(m_{ij}, x_j^w\right)$$

**Caso 2.** Recuperación de un patrón alterado. Un patrón $\tilde{\mathbf{x}}$, que es una versión alterada de un patrón $\mathbf{x}^w$, se le presenta a la memoria $\mathbf{M}$ y se realiza la siguiente operación:

$$\mathbf{M} \lozenge_{\mathrm{B}} \tilde{\mathbf{x}}$$

De nuevo, es resultado es un vector de dimensión $n$, con la $i$-ésima componente dada como:

$$\left(\mathbf{M} \lozenge_B \tilde{\mathbf{x}}\right)_i = \underset{j=1}{\overset{n}{\mathbf{med}}} \mathrm{B}\left(m_{ij}, \tilde{x}_j\right)$$

## 2.3 Estado del arte sobre Memorias Asociativas Bidireccionales

El estado del arte correspondiente al contenido de esta tesis incluye de manera importante los diferentes modelos de memorias asociativas bidireccionales surgidos a partir de 1988, año en que fue creada la primera memoria asociativa bidireccional (BAM) por Kosko [14]. Esta sección consta de dos partes. En la primera parte se describe el funcionamiento, tanto en la fase de aprendizaje como en la de recuperación, de la BAM de Kosko y se incluye un ejemplo ilustrativo. La importancia de este modelo radica en que, además de ser la primera memoria asociativa bidireccional, es la base fundamental teórica que utilizan la mayoría de los modelos siguientes que intentan mejorar la capacidad de recuperación. En la segunda parte se presenta una tabla resumen, tabla 2.3.1, donde se muestran los 38 modelos de memorias asociativas bidireccionales más importantes en orden cronológico.

### 2.3.1 Memoria Asociativa Bidireccional de Kosko.

Bart Kosko, investigador de la *University of Southern California*, propuso en 1988 la *Bidireccional Associative Memory* (BAM) [14] para subsanar la clara desventaja de la autoasociatividad de la memoria Hopfield. La BAM maneja pares de vectores $(A_1,B_1)$, ... $(A_m,B_m)$, donde $A \in \{0, 1\}^n$ y $B \in \{0, 1\}^p$.

Al igual que Austin ensambló dos redes asociativas de Willshaw para diseñar su ADAM [27], Kosko ideó un arreglo de dos memorias Hopfield, y demostró que este diseño es capaz de asociar patrones de manera heteroasociativa.

El funcionamiento de la BAM se ilustra en la figura 2.3.

La matriz **M** es una memoria Hopfield con la única diferencia que la diagonal principal es diferente de cero. $\mathbf{M}^T$ es la matriz transpuesta de **M** que, ahora como entrada, recibe a *B* y la salida será *A*. El proceso bidireccional anteriormente ilustrado continúa hasta que *A* y *B* convergen a una pareja estable ($A_i$, $B_i$).

$$
\begin{aligned}
A &\rightarrow \mathbf{M} \rightarrow B \\
A' &\leftarrow \mathbf{M}^T \leftarrow B \\
A' &\rightarrow \mathbf{M} \rightarrow B' \\
A'' &\leftarrow \mathbf{M}^T \leftarrow B' \\
&\ldots \\
A_i &\rightarrow \mathbf{M} \rightarrow B_i \\
A_i &\leftarrow \mathbf{M}^T \leftarrow B_i
\end{aligned}
$$

**Figura 2.3.1** Funcionamiento de la BAM de Kosko. Se le presenta un patrón de entrada a la matriz de correlación **M** y se obtiene un patrón de salida; posteriormente, se calcula la transpuesta de **M** y se le presenta este patrón de salida, lo que resulta en un nuevo patrón de entrada. Este proceso se repite hasta que el sistema llega a un estado estable en donde no existen cambios en los patrones calculados.

Kosko descubrió que su memoria funcionaba mejor con patrones bipolares que con patrones binarios (a la manera de Hopfield), por tanto: $A \in \{-1, 1\}^n$ y $B \in \{-1, 1\}^p$

Para la codificación de la BAM se superponen las *m* asociaciones sumándolas para formar la matriz de correlación:

$$\mathbf{M} = \sum_i A_i^T B_i \tag{2.3.1}$$

y la memoria dual $\mathbf{M}^T$ que está dada por:

$$\mathbf{M}^T = \sum_i \left(A_i^T B_i\right)^t = \sum_i B_i^T A_i \tag{2.3.2}$$

En el proceso de decodificación, cada neurona $a_i$ que se encuentra en el campo *A* y cada neurona $b_i$ localizada en el campo *B*, de forma independiente y asíncrona, examina la suma de entrada de las neuronas del otro campo, entonces puede o no cambiar su estado si la suma de entrada es mayor, igual o menor que un umbral dado. Si la suma de entrada es igual al umbral, entonces la neurona no cambia su estado. La suma de entrada para $b_j$ es el producto interno columna:

$$A\mathbf{M}^j = \sum_i a_i m_{ij} \tag{2.3.3}$$

donde $\mathbf{M}^j$ es la $j$-ésima columna de $\mathbf{M}$. La suma de entrada para $a_i$ es, de manera similar,

$$B\mathbf{M}_i^T = \sum_j b_j m_{ij} \tag{2.3.4}$$

donde $\mathbf{M}_i$ es la $i$-ésima fila de $\mathbf{M}$. Se toma el 0 como el umbral para todas las neuronas. Las funciones de umbral para $a_i$ y $b_j$ son:

$$a_i = \begin{cases} 1, & \text{si} \quad B\mathbf{M}_i^T > 0 \\ -1, & \text{si} \quad B\mathbf{M}_i^T < 0 \end{cases}$$

$$b_j = \begin{cases} 1, & \text{si} \quad A\mathbf{M}^j > 0 \\ -1, & \text{si} \quad A\mathbf{M}^j < 0 \end{cases}$$

Cuando se le presenta un patrón ($A$, $B$) a la BAM, las neuronas en los campos $A$ y $B$ se prenden o se apagan de acuerdo a la ocurrencia de 1's y 0's en los vectores de estado $A$ y $B$. Las neuronas continúan sus cambios de estado hasta que se alcance un estado estable bidireccional ($A_f$, $B_f$).

El siguiente ejemplo ilustra el funcionamiento de la BAM.

Ejemplo: Se proponen 3 pares de vectores ($m$ = 3):

$A_1$ = (1 1 1 -1 1 -1); $B_1$ = (-1 1 1 1 -1)
$A_2$ = (-1 1 -1 1 1); $B_2$ = (1 1 1 -1 -1)
$A_3$ = (-1 -1 –1 1 –1 1); $B_3$ = (-1 1 -1 1 1)

Por tanto, $n$ = 3 y $p$ = 5.

La matriz de correlación y su transpuesta son:

$$\mathbf{M} = \begin{pmatrix} 1 & -1 & 1 & -1 & 1 & 1 \\ 1 & -1 & 1 & -1 & 1 & 1 \\ 3 & 1 & 3 & -3 & 3 & -1 \\ -1 & 1 & -1 & 1 & -1 & -1 \\ -3 & -1 & -3 & 3 & -3 & 1 \end{pmatrix} \qquad \mathbf{M}^T = \begin{pmatrix} 1 & 1 & 3 & -1 & -3 \\ -1 & -1 & 1 & 1 & -1 \\ 1 & 1 & 3 & -1 & -3 \\ -1 & -1 & -3 & 1 & 3 \\ 1 & 1 & 3 & -1 & -3 \\ 1 & 1 & -1 & -1 & 1 \end{pmatrix}$$

Se le presenta el vector $A_2$ a la memoria de Kosko, entonces, siguiendo el procedimiento descrito en la figura 2.3.1 y utilizando las ecuaciones 2.3.3 y 2.3.4, el resultado es el siguiente:

Con ayuda de la ecuación 2.3.3 se obtiene $B_2$,

$$B_2 = A_2\mathbf{M} = \begin{pmatrix} 6 \\ 6 \\ 10 \\ -6 \\ -10 \end{pmatrix} \qquad \text{Aplicando los umbrales} \qquad B_2 = \begin{pmatrix} 1 \\ 1 \\ 1 \\ -1 \\ -1 \end{pmatrix}$$

con $B_2$ se obtendrá $A_2$' mediante la matriz transpuesta $\mathbf{M}^T$:

$$A_2^{'} = B_2\mathbf{M}^T = \begin{pmatrix} 9 \\ -1 \\ 9 \\ -9 \\ 9 \\ 1 \end{pmatrix} \qquad \text{Aplicando los umbrales} \qquad A_2^{'} = \begin{pmatrix} 1 \\ -1 \\ 1 \\ -1 \\ 1 \\ 1 \end{pmatrix}$$

Ahora utilizando $A_2$' y $\mathbf{M}$ se obtiene $B_2$':

$$B_2^{'} = A_2^{'}\mathbf{M} = \begin{pmatrix} 6 \\ 6 \\ 10 \\ -6 \\ -10 \end{pmatrix} \qquad \text{Aplicando los umbrales} \qquad B_2^{'} = \begin{pmatrix} 1 \\ 1 \\ 1 \\ -1 \\ -1 \end{pmatrix}$$

Comparando $B_2$ con $B_2$' se puede observar que ambos vectores son iguales; por lo tanto, el patrón recuperado es (1 1 1 -1 -1). Se puede concluir que la recuperación realizada por la memoria de Kosko, con el patrón de entrada $A_2$, fue correcta.

Cuando se le presentan los patrones de entrada $A_1$ y $A_3$ a la BAM, los patrones recuperados, $B_1$'' y $B_3$'', son iguales al patrón (1 1 1 -1 -1). Claramente se puede observar que la recuperación no fue correcta en ninguno de los dos casos.

Kosko propone la función de potencial:

$$E(A,B) = -\frac{1}{2}A\mathbf{M}B^T - \frac{1}{2}B\mathbf{M}^T A^T \qquad (2.3.5)$$

como la energía del sistema de la BAM para el estado $(A,B)$. La ecuación 2.3.5 es equivalente a

$$E(A, B) = - A\mathbf{M}B^T \qquad (2.3.6)$$

Kosko comprueba que el proceso de decodificación (proceso bidireccional) siempre alcanza un estado estable, el cual corresponde a un mínimo local de la ecuación 2.3.5. Además, establece que se llega al mínimo local a partir de cualquier matriz de correlación **M**. Se alcanza el estado estable debido a que los cambios en las variables de estado producen un cambio de energía negativo.

En este trabajo, Kosko afirma que la capacidad de almacenamiento de la BAM es el valor mínimo entre las dimensiones de los vectores de los patrones de entrada y de salida, esto es: $\min(n, p)$.

### 2.3.2 Modelos de Memorias Asociativas Bidireccionales a través del tiempo

En esta sección se presentan, en la tabla 2.3.1, los 38 modelos de memorias asociativas bidireccionales más importantes a través del tiempo. La primera columna de la tabla indica el año de creación de la BAM; en la segunda columna aparece el nombre del autor principal y el nombre del modelo (el nombre original en inglés); finalmente, en la tercera columna se describe el método que utilizó cada modelo para intentar mejorar la capacidad de recuperación. En el apéndice B se describe en detalle cada uno de los modelos incluidos en la tabla.

El criterio utilizado para decidir que los 38 modelos de memorias asociativas bidireccionales de la tabla son los más importantes a través del tiempo, radica en el método utilizado por cada modelo. Algunas de ellas son modificaciones de algoritmos propuestos o, en algunas ocasiones, se conjuntan dos métodos para dar vida a uno nuevo. Se tomó la decisión de no mencionar aquellos modelos presentes en la literatura actual que son modificaciones menores de los incluidos en la tabla 2.3.1.

La variedad de conceptos matemáticos y técnicas computacionales usadas por los investigadores es notable. En los modelos de BAM presentados en esta sección están incluidos, entre otros, los siguientes métodos: entrenamiento múltiple y adición *dummy*, técnicas de alto orden, técnicas exponenciales, codificación Householder, algoritmo *Optimal Gradient Descent,* conexión asimétrica, aprendizaje basado en el *perceptron* y eliminación de estados espurios.

**Tabla 2.3.1 Modelos de BAM a través del tiempo**

| Año | Autor y Nombre | Característica distintiva |
|---|---|---|
| 1988 | Kosko, *BAM* [14] | Primera memoria asociativa bidireccional |
| 1988 | Haines *et. al.*, *BAM no homogénea* [16] | Se introducen los umbrales diferentes de cero |
| 1989 | Wang Y *et al*, *Multiple Training (MT)* [28] | Se entrena la BAM con la misma pareja de patrones múltiples ocasiones para intentar que, al menos, esta pareja sea recuperada |

| | | |
|---|---|---|
| 1989 | Tai *et al*, *High-Order Bidirectional Associative Memory (HOBAM)* [29] | En la etapa de codificación de la HOBAM de orden *k*, se utiliza la operación de elevar al exponente *k* el producto interno de los vectores de entrada |
| 1990 | Jeng *et al*, *Exponencial BAM (EBAM)* [30] | La regla de actualización utiliza la operación exponencial que contiene como base un número $\alpha$>1 y como exponente el producto interno de los vectores de entrada |
| 1990 | Wang Y *et al*, *MT and dummy addition* [31] | Utiliza de nuevo el entrenamiento múltiple con una variante: la extensión dummy de los vectores |
| 1990 | Wang Y *et al* [32] | Especifica el número mínimo de veces de entrenamiento de una asociación para asegurar que el correspondiente par sea recuperado en el proceso de entrenamiento múltiple |
| 1990 | Simpson, *Higher Ordered BAM* [33] | La intra e interconexión de las diferentes capas de los heterocorrelacionadores |
| 1991 | Wang Y *et al*, *Linear Programming/Multiple Training (LP/MT)* [19] | Mediante técnicas de programación lineal se calcula el número de veces de entrenamiento de una asociación para asegurar que el correspondiente par sea recuperado en el proceso de entrenamiento múltiple |
| 1991 | Leung *et al*, *Codificación Householder* [17] | Se aplica la transformada Householder en la etapa de codificación |
| 1991 | Srinivasan *et al*, *Unlearning* [34] | Se crea la matriz de correlación cancelando los estados espurios (se induce a que algunos patrones no se aprendan) |
| 1992 | Wang W *et al*<br>*Modified EBAM (MEBAM)* [35] | Agrega un término de autocorrelación al exponente |
| 1992 | Jeng *et al*<br>*Stable High order BAM ( SHBAM)* [36] | Modelo general de correlación para asegurar estabilidad en la HOBAM y en la EBAM |
| 1992 | Yu *et al*, *Generalized Bidirectional Associative Memory* (GBAM) [37] | Pondera los patrones fundamentales de manera que cumplan con la condición de continuidad propuesta por Kosko, en lugar de buscar que los patrones sean estados estables |
| 1993 | Lee *et al*, *Correlation Significance (CS)* [38] | Similar al LP/MT, se generan ponderaciones. Se utiliza el método del gradiente descendiente. La función de error (ponderaciones) se actualiza en cada paso de la recuperación |
| 1993 | Perfetti<br>*Optimal Gradient Descent Learning* [39] | Utiliza el método del gradiente descendiente de manera que se logren maximizar los radios de atracción de cada uno de los patrones entrenados |
| 1993 | Leung, *Enhanced Householder Coding Algorithm (EHCA)* [40] | Fusiona en una sola matriz las dos matrices del Householder |
| 1993 | Leung, *Adaptative Ho-Kashyap Bidirectional Learning ( AHKBL*) [41] | Propone una nueva regla de aprendizaje para la BAM: Adaptive Ho-Kashyap Bidirectional Learning (AHKBL) |
| 1994 | Khorasani *et al*, *BAM modificada* [43] | Utiliza la técnica de *successive over-relaxation* para encontrar pesos y los umbrales. Su objetivo no es minimizar la función de energía, sino generalizar el aprendizaje de memorias autoasociativas basadas en Hopfield |
| 1994 | Wang C *et al*, Análisis de la eBAM implementada con circuitos VLSI [44] | Se realiza el análisis de la eBAM implementada con circuitos VLSI, con respecto a su radio de atracción y su capacidad de almacenamiento. |
| 1994 | Xu *et al*, *Asymmetric Bidirectional Associative Memory (ABAM)* [45] | Propone pesos de interconexión asimétricos para la matriz de correlación. Además, se crea un operador de interpolación en la codificación para asegurar que todos los patrones sean estados estables si son linealmente independientes |
| 1994 | Hattori *et al*, *Pseudo-Relaxation Learning Algorithm (PRLAB)* 46] | Utiliza el PRLAB para encontrar los pesos y los umbrales de la matriz de correlación |

| | | |
|---|---|---|
| 1994 | Leung, *Bidirectional Learning ( BL)* [48] | Se basa en la idea del *perceptron* para generar una matriz de conexiones |
| 1994 | Hu *et al*, *Parallel Orthogonalization Based BAM (POBAM) y Unilateral Orthogonalization Based BAM (UOBAM)* [49] | Se proponen dos tipos de codificación basados en técnicas de ortogonalización: el POBAM, que tiene conexiones simétricas, y el UOBAM, con conexiones asimétricas |
| 1994 | Wang Z, *Linear and No Linear BAM* [50] | Se presentan dos diseños de BAM, una lineal y otra no lineal. Ambas utilizan una matriz óptima diferente a una matriz de correlación |
| 1994 | Sarkar, *Three Layer BAM (TLBAM)* [51] | Se propone una BAM de tres capas, en la que se calcula la cantidad exacta de ruido en la recuperación |
| 1996 | Osana *et al*, *Chaotic Bidirectional Associative Memory (CBAM)* [52] | Utiliza neuronas caóticas en la etapa de decodificación, y permite asociaciones de uno a muchos. |
| 1997 | Chen *et al* *Improved Exponential BAM (IEBAM)* [53] | La IEBAM fusiona los dos exponentes de la MEBAM en uno solo, lo que permite evitar el requisito de continuidad |
| 1997 | Haryono *et al*, *Bipolar-Orthogonal Augmentation Method (BOAM)* [54] | Al igual que Wang Y (1990) se aumentan elementos dummy, la única diferencia es que los elementos de Wang eran binarios y los de Haryono son bipolares |
| 1997 | Araújo *et al*, *Unlearning of Spurious Attractors (USA) and Bidirectional Delta Rule (BDR)* [55] | El USA elimina los atractores espurios y el BDR actualiza los pesos de la matriz de correlación bidireccionalmente mientras existan atractores espurios |
| 1999 | Ritter *et al*, *Morphological BAM (MBAM)* [57] | Se basan en las operaciones morfológicas de dilatación y erosión, y son *one-shot* |
| 2000 | Wu *et al*, *Feedforward BAM* [58] | Se diseñan dos BAM feedforward, una compuesta por los patrones de entrada y un vector intermedio, y la otra, con los patrones de salida y el mismo vector; además, es *one-shot* |
| 2001 | Lenze, *Improved BL (IBL)* [59] | La condición para el buen funcionamiento del BL era que los patrones fueran linealmente separables; IBL utiliza la traslación y la dilatación para evadir esa condición |
| 2001 | Eom *et al*, *EBAM ponderada* [60] | Basada en la eBAM se calcula el radix que garantiza la atracción dentro de cierto radio. Las ponderaciones utilizadas para obtener el radix se calculan mediante el algoritmo *Pseudo-Relaxation* |
| 2003 | Lee *et al* [63] | Se propone el concepto de bola de Hamming de orden *q* |
| 2005 | Zheng *et al*, *Optimal Gradient Descent Algorithm (OGDA)* [64] | Utiliza el método del aumento *dummy* para la corrección de los patrones recuperados; además, agranda el radio de atracción mediante el algoritmo del gradiente descendiente óptimo |
| 2005 | Shen *et al* [18] | Mediante un algoritmo genético se pondera la matriz de correlación |
| 2006 | Chartier [65] | Permite el aprendizaje tanto de patrones binarios como de patrones en escala de grises |

En la figura 2.3.2 podemos observar los resultados gráficos de la capacidad de recuperación de algunos de los modelos presentados en la tabla anterior. Las gráficas de los modelos de Wang (1991), Jeng, Zheng y Chartier nos indican que la recuperación de patrones decrece cuando aumenta el número de parejas entrenadas. Sin embargo, el nuevo modelo de Memoria Asociativa Bidireccional propuesto en este trabajo de tesis tiene el comportamiento indicado por la flecha que sobresale en negro; esto es, la BAM Alfa-Beta recupera de manera correcta todos los patrones entrenados, sin importar que el número de éstos aumente. La gráfica resultante del modelo de Wu sugiere que para que este modelo pueda recuperar todos los patrones entrenados, la distancia de Hamming entre ellos debe ser menor o igual a 4. Las características de los patrones entrenados, como: distancia de Hamming, ortogonalidad, independencia lineal, entre otras, no limitan el buen

funcionamiento de la BAM Alfa-Beta; el nuevo modelo siempre presentará recuperación correcta de todos los patrones del conjunto fundamental.

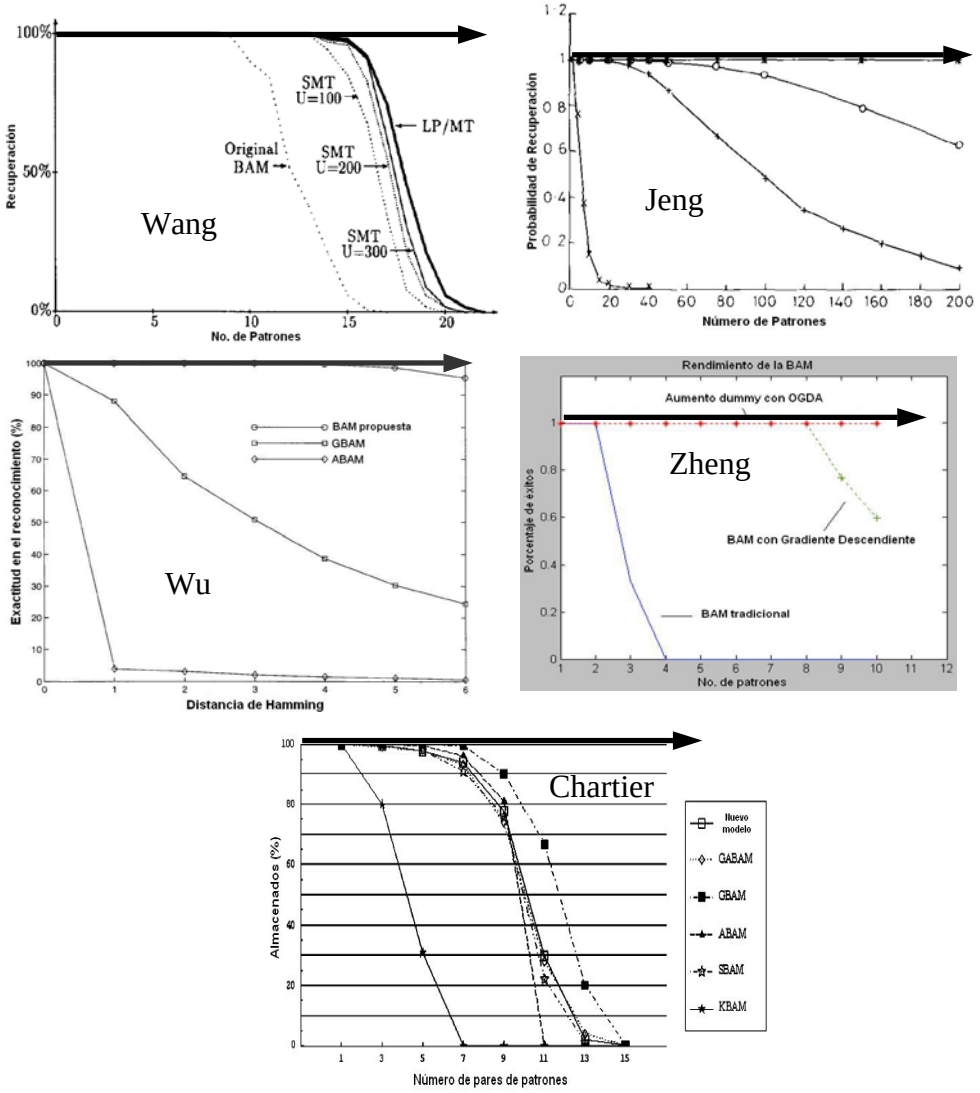



**Figura 2.3.2** Gráficas de los resultados de la capacidad de recuperación de los modelos de Wang (1991), Jeng, Wu, Zheng y Chartier (mostrados en la tabla 2.3.1). La flecha que sobresale en negro, indica el comportamiento de la BAM Alfa-Beta ante la capacidad de recuperación, lo que nos sugiere la recuperación correcta de todos los patrones entrenados.

En al Apéndice B se describen detalladamente cada uno de los 38 modelos de Memorias Asociativas Bidireccionales que se presentaron en la Tabla 2.3.1.

# CAPÍTULO 3

# Marco Teórico

Las Memorias Asociativas Alfa-Beta son la base fundamental del modelo de Memoria Asociativa Bidireccional propuesto en este trabajo de tesis.

En este capítulo se presenta el fundamento teórico que sustenta a las memorias asociativas Alfa-Beta tal como aparece en [15]; para ello, se presentan las definiciones de las operaciones $\alpha$ y $\beta$, las operaciones matriciales utilizando estas operaciones originales, y se describen las fases de aprendizaje y recuperación de la memorias heteroasociativas y autoasociativas Alfa-Beta, tanto *max* como *min*.

La numeración de los Lemas y Teoremas que se presentan en este capítulo, corresponde a la numeración original que aparece en [15].

## 3.1 Memorias Asociativas Alfa-Beta

Las memorias Alfa-Beta utilizan máximos y mínimos, y dos operaciones binarias originales $\alpha$ y $\beta$ de las cuales heredan el nombre.

Para la definición de las operaciones binarias $\alpha$ y $\beta$ se deben especificar los conjuntos $A$ y $B$, los cuales son:

$$A = \{0, 1\} \quad \text{y} \quad B = \{0, 1, 2\}$$

La operación binaria $\alpha$: $A \times A \rightarrow B$ se define como se muestra en la Tabla 3.1.

**Tabla 3.1** Operación binaria $\alpha$: $A \times A \rightarrow B$

| $x$ | $y$ | $\alpha(x, y)$ |
|---|---|---|
| 0 | 0 | 1 |
| 0 | 1 | 0 |
| 1 | 0 | 2 |
| 1 | 1 | 1 |

La operación binaria $\beta$: $B \times A \rightarrow A$ se define como se muestra en la Tabla 3.2

Los conjuntos *A* y *B*, las operaciones binarias $\alpha$ y $\beta$ junto con los operadores $\wedge$ (mínimo) y $\vee$ (máximo) usuales conforman el sistema algebraico (A, B, $\alpha$, $\beta$, $\wedge$, $\vee$) en el que están inmersas las memorias asociativas Alfa-Beta.

**Tabla 3.2** Operación binaria $\beta$: $B \times A \rightarrow A$

| *x* | *y* | $\beta(x, y)$ |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 1 |
| 2 | 0 | 1 |
| 2 | 1 | 1 |

Se requiere la definición de cuatro operaciones matriciales, de las cuales se usarán sólo 4 casos particulares:

Operación ***αmax***: $P_{mxr} \nabla_{\alpha} Q_{rxn} = \left[ f_{ij}^{\alpha} \right]_{mxn}$, donde $f_{ij}^{\alpha} = \bigvee_{k=1}^{r} \alpha(p_{ik}, q_{kj})$

Operación ***βmax***: $P_{mxr} \nabla_{\beta} Q_{rxn} = \left[ f_{ij}^{\beta} \right]_{mxn}$, donde $f_{ij}^{\beta} = \bigvee_{k=1}^{r} \beta(p_{ik}, q_{kj})$

Operación ***αmin***: $P_{mxr} \Delta_{\alpha} Q_{rxn} = \left[ h_{ij}^{\alpha} \right]_{mxn}$, donde $h_{ij}^{\alpha} = \bigwedge_{k=1}^{r} \alpha(p_{ik}, q_{kj})$

Operación ***βmin***: $P_{mxr} \Delta_{\beta} Q_{rxn} = \left[ h_{ij}^{\beta} \right]_{mxn}$, donde $h_{ij}^{\beta} = \bigwedge_{k=1}^{r} \beta(p_{ik}, q_{kj})$

El siguiente lema muestra los resultados obtenidos al utilizar las operaciones que involucran al operador binario $\alpha$ con las componentes de un vector columna y un vector fila dados.

**Lema 2.1.** Sean $\mathbf{x} \in A^n$ y $\mathbf{y} \in A^m$; entonces $\mathbf{y} \nabla_{\alpha} \mathbf{x}^t$ es una matriz de dimensiones *m* x *n*, y además se cumple que: $\mathbf{y} \nabla_{\alpha} \mathbf{x}^t = \mathbf{y} \Delta_{\alpha} \mathbf{x}^t$.

Demostración.

$$\mathbf{y}\nabla_\alpha \mathbf{x}^t = \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{pmatrix} \nabla_\alpha \left(x_1, x_2, \ldots, \mathrm{x_n}\right)$$

$$= \begin{pmatrix} \bigvee_{k=1}^{1} \alpha(y_1, x_1) & \bigvee_{k=1}^{1} \alpha(y_1, x_2) & \ldots & \bigvee_{k=1}^{1} \alpha(y_1, x_n) \\ \bigvee_{k=1}^{1} \alpha(y_2, x_1) & \bigvee_{k=1}^{1} \alpha(y_2, x_2) & \ldots & \bigvee_{k=1}^{1} \alpha(y_2, x_n) \\ \vdots & \vdots & \ldots & \vdots \\ \bigvee_{k=1}^{1} \alpha(y_m, x_1) & \bigvee_{k=1}^{1} \alpha(y_m, x_2) & \ldots & \bigvee_{k=1}^{1} \alpha(y_m, x_n) \end{pmatrix}_{mxn}$$

$$= \begin{pmatrix} \alpha(y_1, x_1) & \alpha(y_1, x_2) & \ldots & \alpha(y_1, x_n) \\ \alpha(y_2, x_1) & \alpha(y_2, x_2) & \ldots & \alpha(y_2, x_n) \\ \vdots & \vdots & \ldots & \vdots \\ \alpha(y_m, x_1) & \alpha(y_m, x_2) & \ldots & \alpha(y_m, x_n) \end{pmatrix}_{mxn}$$

$$= \begin{pmatrix} \bigwedge_{k=1}^{1} \alpha(y_1, x_1) & \bigwedge_{k=1}^{1} \alpha(y_1, x_2) & \ldots & \bigwedge_{k=1}^{1} \alpha(y_1, x_n) \\ \bigwedge_{k=1}^{1} \alpha(y_2, x_1) & \bigwedge_{k=1}^{1} \alpha(y_2, x_2) & \ldots & \bigwedge_{k=1}^{1} \alpha(y_2, x_n) \\ \vdots & \vdots & \ldots & \vdots \\ \bigwedge_{k=1}^{1} \alpha(y_m, x_1) & \bigwedge_{k=1}^{1} \alpha(y_m, x_2) & \ldots & \bigwedge_{k=1}^{1} \alpha(y_m, x_n) \end{pmatrix}_{mxn}$$

$$= \mathbf{y}\Delta_\alpha \mathbf{x}^t$$

En efecto, resulta que $\mathbf{y} \nabla_\alpha \mathbf{x}^t$ es una matriz de dimensiones *m* x *n*, y que $\mathbf{y} \nabla_\alpha \mathbf{x}^t = \mathbf{y} \Delta_\alpha \mathbf{x}^t$.

Dado el resultado del lema anterior, es conveniente escoger un símbolo único, digamos el símbolo $\otimes$, que represente a las dos operaciones $\nabla_\alpha$ y $\Delta_\alpha$ cuando se opera un vector columna de dimensión *m* con un vector fila de dimensión *n*:

$$\mathbf{y} \nabla_\alpha \mathbf{x}^t = \mathbf{y} \otimes \mathbf{x}^t = \mathbf{y} \Delta_\alpha \mathbf{x}^t$$

La *ij*-ésima componente de la matriz está $\mathbf{y} \otimes \mathbf{x}^t$ dada por:

$$[\mathbf{y} \otimes \mathbf{x}^t]_{ij} = \alpha(y_i, x_j)$$

Dado un índice de asociación $\mu$, la expresión anterior indica que la *ij*-ésima componente de la matriz $\mathbf{y}^\mu \otimes (\mathbf{x}^\mu)^t$ se expresa de la siguiente manera:

$$[\mathbf{y}^\mu \otimes (\mathbf{x}^\mu)^t]_{ij} = \alpha\left(y_i^\mu, x_j^\mu\right)$$

Ahora se analizará el caso en el que se opera una matriz de dimensiones *m* x *n* con un vector columna de dimensión n usando las operaciones$\nabla_\beta$ y $\Delta_\beta$. En los lemas 2.2 y 2.3 se

obtiene la forma que exhibirán las *i*-ésimas componentes de los vectores columna resultantes de dimensión *m*, a partir de ambas operaciones $\nabla_\beta$ y $\Delta_\beta$.

**Lema 2.2** Sean $\mathbf{x} \in A^n$ y $\mathbf{P}$ una matriz de dimensiones *m* x *n*. La operación $\mathbf{P}_{mxn}\ \nabla_\beta\ \mathbf{x}$ da como resultado un vector columna de dimensión *m*, cuya *i*-ésima componente tiene la siguiente forma: $\left(\mathbf{P}_{mxn}\nabla_\beta \mathbf{x}\right)_i = \bigvee_{j=1}^{n} \beta\left(p_{ij}, x_j\right)$

Demostración.-

$$\mathbf{P}_{mxn}\nabla_\beta \mathbf{x} = \begin{pmatrix} p_{11} & p_{12} & \cdots & p_{1n} \\ p_{21} & p_{22} & \cdots & p_{2n} \\ \vdots & \vdots & \cdots & \vdots \\ p_{m1} & p_{m2} & \cdots & p_{mn} \end{pmatrix} \nabla_\beta \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix}$$

$$\mathbf{P}_{mxn}\nabla_\beta \mathbf{x} = \begin{pmatrix} \beta(p_{11}, x_1) \vee \beta(p_{12}, x_2) \vee \ldots \vee \beta(p_{1n}, x_n) \\ \beta(p_{21}, x_1) \vee \beta(p_{22}, x_2) \vee \ldots \vee \beta(p_{2n}, x_n) \\ \vdots \\ \beta(p_{m1}, x_1) \vee \beta(p_{m2}, x_2) \vee \ldots \vee \beta(p_{mn}, x_n) \end{pmatrix} = \begin{pmatrix} \bigvee_{j=1}^{n} \beta(p_{1j}, x_j) \\ \bigvee_{j=1}^{n} \beta(p_{2j}, x_j) \\ \vdots \\ \bigvee_{j=1}^{n} \beta(p_{mj}, x_j) \end{pmatrix}$$

Se obtiene un vector columna de dimensión *m* cuya *i*-ésima componente es

$$\left(\mathbf{P}_{mxn}\nabla_\beta \mathbf{x}\right)_i = \bigvee_{j=1}^{n} \beta\left(p_{ij}, x_j\right)$$

**Lema 2.3** Sean $\mathbf{x} \in A^n$ y $\mathbf{P}$ una matriz de dimensiones *m* x *n*. La operación $\mathbf{P}_{mxn}\ \Delta_\beta\ \mathbf{x}$ da como resultado un vector columna de dimensión *m*, cuya *i*-ésima componente tiene la siguiente forma: $\left(\mathbf{P}_{mxn}\Delta_\beta \mathbf{x}\right)_i = \bigwedge_{j=1}^{n} \beta\left(p_{ij}, x_j\right)$

Demostración

$$\mathbf{P}_{mxn}\Delta_\beta \mathbf{x} = \begin{pmatrix} p_{11} & p_{12} & \cdots & p_{1n} \\ p_{21} & p_{22} & \cdots & p_{2n} \\ \vdots & \vdots & \cdots & \vdots \\ p_{m1} & p_{m2} & \cdots & p_{mn} \end{pmatrix} \Delta_\beta \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix}$$

$$\mathbf{P}_{mxn}\Delta_\beta \mathbf{x} = \begin{pmatrix} \beta(p_{11}, x_1) \wedge \beta(p_{12}, x_2) \wedge \ldots \wedge \beta(p_{1n}, x_n) \\ \beta(p_{21}, x_1) \wedge \beta(p_{22}, x_2) \wedge \ldots \wedge \beta(p_{2n}, x_n) \\ \vdots \\ \beta(p_{m1}, x_1) \wedge \beta(p_{m2}, x_2) \wedge \ldots \wedge \beta(p_{mn}, x_n) \end{pmatrix} = \begin{pmatrix} \bigwedge_{j=1}^{n} \beta(p_{1j}, x_j) \\ \bigwedge_{j=1}^{n} \beta(p_{2j}, x_j) \\ \vdots \\ \bigwedge_{j=1}^{n} \beta(p_{mj}, x_j) \end{pmatrix}$$

Se obtiene un vector columna de dimensión *m* cuya *i*-ésima componente es

$$\left(\mathbf{P}_{mxn}\Delta_{\beta}\mathbf{x}\right)_i = \bigwedge_{j=1}^{n}\beta\left(p_{ij}, x_j\right)$$

## 3.2 Memorias Heteroasociativas Alfa-Beta

Se tienen dos tipos de memorias heteroasociativas Alfa-Beta: tipo **V** y tipo **Λ**. En la generación de ambos tipos de memorias se usará el operador ⊗ el cual tiene la siguiente forma:

$$\left[\mathbf{y}^{\mu}\otimes\left(\mathbf{x}^{\mu}\right)^{t}\right]_{ij} = \alpha\left(y_i^{\mu}, x_j^{\mu}\right);\ \mu\in\{1, 2, ..., p\},\ i\in\{1, 2, ..., m\},\ j\in\{1, 2, ..., n\}$$

### Algoritmo Memorias Alfa-Beta tipo V

### Fase de Aprendizaje

**Paso 1.** Para cada $\mu = 1, 2, ..., p$, a partir de la pareja $(\mathbf{x}^{\mu}, \mathbf{y}^{\mu})$ se construye la matriz

$$\left[\mathbf{y}^{\mu}\otimes\left(\mathbf{x}^{\mu}\right)^{t}\right]_{mxn}$$

**Paso 2.** Se aplica el operador binario máximo ∨ a las matrices obtenidas en el paso 1:

$$\mathbf{V} = \bigvee_{\mu=1}^{p}[\mathbf{y}^{\mu}\otimes(\mathbf{x}^{\mu})^{t}]$$

La entrada *ij*-ésima está dada por la siguiente expresión:

$$v_{ij} = \bigvee_{\mu=1}^{p}\alpha(y_i^{\mu}, x_j^{\mu})$$

### Fase de Recuperación

Se presenta un patrón $\mathbf{x}^{\omega}$, con $\omega\in\{1, 2, ..., p\}$, a la memoria heteroasociativa $\alpha\beta$ tipo **V** y se realiza la operación $\Delta_{\beta}$: $\mathbf{V}\ \Delta_{\beta}\ \mathbf{x}^{\omega}$.

Dado que las dimensiones de la matriz **V** son de *m* x *n* y $\mathbf{x}^{\omega}$ es un vector columna de dimensión *n*, el resultado de la operación anterior debe ser un vector columna de dimensión *m*, cuya *i*-ésima componente es:

$$\left(\mathbf{V}\Delta_{\beta}\mathbf{x}^{\omega}\right)_i = \bigwedge_{j=1}^{n}\beta\left(v_{ij}, x_j^{\mu}\right)$$

### Algoritmo Memorias Alfa-Beta tipo $\Lambda$

#### Fase de Aprendizaje

**Paso 1.** Para cada $\mu = 1, 2, ..., p$, a partir de la pareja $(\mathbf{x}^{\mu}, \mathbf{y}^{\mu})$ se construye la matriz

$$\left[\mathbf{y}^{\mu} \otimes \left(\mathbf{x}^{\mu}\right)^{t}\right]_{mxn}$$

**Paso 2.** Se aplica el operador binario mínimo $\wedge$ a las matrices obtenidas en el paso 1:

$$\mathbf{\Lambda} = \bigwedge_{\mu=1}^{p} [\mathbf{y}^{\mu} \otimes (\mathbf{x}^{\mu})^{t}]$$

La entrada *ij*-ésima está dada por la siguiente expresión:

$$\lambda = \bigwedge_{\mu=1}^{p} \alpha(y_i^{\mu}, x_j^{\mu})$$

#### Fase de Recuperación

Se presenta un patrón $\mathbf{x}^{\omega}$, con $\omega \in \{1, 2, ..., p\}$, a la memoria heteroasociativa $\alpha\beta$ tipo $\mathbf{\Lambda}$ y se realiza la operación $\nabla_{\beta}$: $\mathbf{V} \nabla_{\beta} \mathbf{x}^{\omega}$.

Dado que las dimensiones de la matriz $\mathbf{\Lambda}$ son de *m* x *n* y $\mathbf{x}^{\omega}$ es un vector columna de dimensión *n*, el resultado de la operación anterior debe ser un vector columna de dimensión *m*, cuya *i*-ésima componente es:

$$\left(\mathbf{\Lambda} \nabla_{\beta} \mathbf{x}^{\omega}\right)_i = \bigvee_{j=1}^{n} \beta\left(\lambda_{ij}, x_j^{\mu}\right)$$

## 3.3 Memorias Autoasociativas Alfa-Beta

Si a una memoria heteroasociativa se le impone la condición de que $\mathbf{y}^{\mu} = \mathbf{x}^{\mu}$ $\forall \mu \in \{1, 2, ..., p\}$ entonces, deja de ser heteroasociativa y ahora se le denomina autoasociativa.

A continuación se enlistan algunas de las características de las memorias autoasociativas Alfa-Beta :

1. El conjunto fundamental toma la forma $\{(\mathbf{x}^{\mu},\mathbf{x}^{\mu}) \mid \mu = 1, 2, ..., p\}$
2. Los patrones fundamentales de entrada y salida son de la misma dimensión; denotémosla por *n*.
3. La memoria es una matriz cuadrada, para ambos tipos, $\mathbf{V}$ y $\mathbf{\Lambda}$. Si $\mathbf{x}^{\mu} \in A^{n}$ entonces

$$\mathbf{V} = \left[v_{ij}\right]_{nxn} \text{ y } \Lambda = \left[\lambda_{ij}\right]_{nxn}$$

### 3.3.1 Memorias Autoasociativas Alfa-Beta tipo V

Las fases de aprendizaje y recuperación son similares a las memorias heteroasociativas Alfa-Beta.

#### Fase de Aprendizaje

**Paso 1.** Para cada $\mu = 1, 2, ..., p$, a partir de la pareja $(\mathbf{x}^{\mu},\mathbf{x}^{\mu})$ se construye la matriz

$$\left[\mathbf{x}^{\mu} \otimes (\mathbf{x}^{\mu})^{t}\right]_{nxn}$$

**Paso 2.** Se aplica el operador binario máximo $\mathbf{V}$ a las matrices obtenidas en el paso 1:

$$\mathbf{V} = \bigvee_{\mu=1}^{p} \left[\mathbf{x}^{\mu} \otimes (\mathbf{x}^{\mu})^{t}\right]$$

La entrada *ij*-ésima de la memoria está dada así:

$$v_{ij} = \bigvee_{\mu=1}^{p} \alpha\left(x_i^{\mu}, x_j^{\mu}\right)$$

y de acuerdo con que $\alpha$: $A$ x $A \rightarrow B$, se tiene que $v_{ij} \in B$, $\forall i \in \{1, 2, ..., n\}$. $\forall j \in \{1, 2, ..., n\}$.

**Fase de Recuperación.** La fase de recuperación de las memorias autoasociativas Alfa-Beta tipo V tiene dos casos posibles. En el primer caso el patrón de entrada es un patrón fundamental; es decir, la entrada es un patrón $\mathbf{x}^{\omega}$, con $\omega \in \{1, 2, ..., p\}$. En el segundo caso, el patrón de entrada NO es un patrón fundamental, sino la versión distorsionada de por lo menos uno de los patrones fundamentales; lo anterior significa que si el patrón de entrada es $\tilde{\mathbf{x}}$, debe existir al menos un valor de índice $\omega \in \{1, 2, ..., p\}$, que corresponde al patrón fundamental respecto del cual $\tilde{\mathbf{x}}$ es una versión alterada con alguno de los tres tipos de ruido: aditivo, sustractivo o mezclado.

**CASO 1: Patrón fundamental.** Se presenta a un patrón $\mathbf{x}^{\omega}$, con $\omega \in \{1, 2, ..., p\}$ a la memoria autoasociativa Alfa-Beta tipo V y se realiza la operación $\Delta_{\beta}$:

$$\mathbf{V}\ \Delta_{\beta}\ \mathbf{x}^{\omega}$$

El resultado de la operación anterior será el vector columna de dimensión *n*.

$$\left(\mathbf{V}\Delta_{\beta}\mathbf{x}^{\omega}\right)_i = \bigwedge_{j=1}^{n} \beta(v_{ij}, x_j^{\omega})$$

$$\left(\mathbf{V}\Delta_\beta \mathbf{x}^\omega\right)_i = \bigwedge_{j=1}^{n} \beta\left\{\left[\bigvee_{\mu=1}^{p} \alpha(x_i^\mu, x_j^\mu)\right], x_j^\omega\right\}$$

**CASO 2: Patrón alterado.** Se presenta el patrón binario $\tilde{\mathbf{x}}$ (patrón alterado de algún patrón fundamental $\mathbf{x}^\omega$) que es un vector columna de dimensión *n*, a la memoria autoasociativa Alfa-Beta tipo V y se realiza la operación

$$\mathbf{V}\Delta_\beta \tilde{\mathbf{x}}$$

Al igual que en el caso 1, el resultado de la operación anterior es un vector columna de dimensión *n*, cuya *i*-ésima componente se expresa de la siguiente manera:

$$\left(\mathbf{V}\Delta_\beta \tilde{\mathbf{x}}\right)_i = \bigwedge_{j=1}^{n} \beta(v_{ij}, \tilde{x}_j)$$

$$\left(\mathbf{V}\Delta_\beta \tilde{\mathbf{x}}\right)_i = \bigwedge_{j=1}^{n} \beta\left\{\left[\bigvee_{\mu=1}^{p} \alpha(x_i^\mu, x_j^\mu)\right], \tilde{x}_j\right\}$$

**Lema 4.27.** *Una memoria autoasociativa Alfa-Beta tipo V tiene únicamente unos en la diagonal principal*.

**Demostración**. La *ij*-ésima entrada de una memoria autoasociativa Alfa-Beta tipo **V** está dada por $v_{ij} = \bigvee_{\mu=1}^{p} \alpha\left(x_i^\mu, x_j^\mu\right)$. Las entradas de la diagonal principal se obtienen de la expresión anterior haciendo *i* = *j*:

$$v_{ii} = \bigvee_{\mu=1}^{p} \alpha\left(x_i^\mu, x_i^\mu\right), \quad \forall i \in \{1,2,...n\}$$

Por la propiedad de la tabla se tiene que $\alpha(x_i^\mu, x_i^\mu) = 1$, por lo que la expresión anterior se transforma en:

$$v_{ii} = \bigvee_{\mu=1}^{p} (1) = 1, \quad \forall i \in \{1,2,...n\}$$

**Teorema 4.28** Una memoria autoasociativa Alfa-Beta tipo V recupera de manera perfecta el conjunto fundamental completo; además, tiene máxima capacidad de aprendizaje.

Demostración. Sea $\omega = \{1, 2, ..., p\}$ arbitrario. De acuerdo con el lema 4.27, para cada $i \in \{1, ..., n\}$ escogida arbitrariamente

$$v_{ii} = 1 = \alpha(x_i^{\omega}, x_i^{\omega})$$

Es decir, para $i \in \{1, ..., n\}$ escogida arbitrariamente, $\exists\, j_0 = i \in \{1, ..., n\}$ que cumple con:

$$v_{ij_0} = \alpha(x_i^{\omega}, x_{j_0}^{\omega})$$

Por lo tanto, de acuerdo con el Teorema 4.7

$$\mathbf{V} \Delta_{\beta} \mathbf{x}^{\omega} = \mathbf{x}^{\omega}, \forall \omega \in \{1, 2, ..., p\}$$

Esto significa que la memoria autoasociativa Alfa-Beta tipo V recupera de manera perfecta el conjunto fundamental completo.

Además, en la demostración de este Teorema, en ningún momento aparece restricción alguna sobre *p*, que es la cardinalidad del conjunto fundamental; y esto quiere decir que el conjunto fundamental puede crecer tanto como se quiera. La consecuencia directa es que el número de patrones que puede aprender una memoria autoasociativa Alfa-Beta tipo V, con recuperación perfecta, es máximo.

El Teorema 4.28 se puede enunciar desde un enfoque matricial de la siguiente manera: dado que para cada asociación ($\mathbf{x}^{\omega}$, $\mathbf{x}^{\omega}$) del conjunto fundamental de una memoria autoasociativa Alfa-Beta V, se cumple que cada fila de la matriz V - $\mathbf{x}^{\omega} \otimes (\mathbf{x}^{\omega})^t$ contiene una entrada cero, entonces de la memoria V recupera el conjunto completo de patrones fundamentales en forma perfecta.

**Teorema 4.30** Sea $\{(\mathbf{x}^{\mu},\mathbf{x}^{\mu}) \mid \mu = 1, 2, ..., p\}$ el conjunto fundamental de una memoria autoasociativa Alfa-Beta representada por V, y sea $\tilde{\mathbf{x}} \in A^n$ un patrón alterado con ruido aditivo respecto a algún patrón fundamental $\mathbf{x}^{\omega}$ con $\omega \in \{1, 2, ..., p\}$. Si se presenta $\tilde{\mathbf{x}}$ a la memoria V como entrada, y si además para cada $i \in \{1, ..., n\}$ se cumple la condición de que $\exists j = j_0 \in \{1, ..., n\}$, el cual depende de $\omega$ y de $i$ tal que $v_{ij_0} \leq \alpha(x^{\omega}, \tilde{x}_{j_0})$, entonces la recuperación $\mathbf{V}\Delta_{\beta}\tilde{\mathbf{x}}$ es perfecta, es decir, $\mathbf{V}\Delta_{\beta}\tilde{\mathbf{x}} = \mathbf{x}^{\omega}$

**Demostración.-** Por hipótesis se tiene que $\mathbf{y}^{\mu} = \mathbf{x}^{\mu}\ \forall \mu \in \{1, 2, ..., p\}$ y, por consiguiente, $m = n$. Al establecer estas dos condiciones en el Teorema 4.13 (Ver referencia), se obtiene el resultado: $\mathbf{V}\Delta_{\beta}\tilde{\mathbf{x}} = \mathbf{x}^{\omega}$

El Teorema 4.30 nos dice que las memorias autoasociativas Alfa-Beta tipo V son inmunes a cierta cantidad de ruido aditivo.

### 3.3.2 Memorias autoasociativas Alfa-Beta Λ

## Fase de Aprendizaje

**Paso 1.** Para cada $\mu = 1, 2, ..., p$, a partir de la pareja $(\mathbf{x}^{\mu},\mathbf{x}^{\mu})$ se construye la matriz

$$\left[\mathbf{x}^{\mu} \otimes (\mathbf{x}^{\mu})^{t}\right]_{nxn}$$

**Paso 2.** Se aplica el operador binario máximo $\boldsymbol{\Lambda}$ a las matrices obtenidas en el paso 1:

$$\boldsymbol{\Lambda} = \bigwedge_{\mu=1}^{p} \left[\mathbf{x}^{\mu} \otimes (\mathbf{x}^{\mu})^{t}\right]$$

La entrada *ij*-ésima de la memoria está dada así:

$$\lambda_{ij} = \bigwedge_{\mu=1}^{p} \alpha\left(x_i^{\mu}, x_j^{\mu}\right)$$

y de acuerdo con que $\alpha$: $A$ x $A \rightarrow B$, se tiene que $\lambda_{ij} \in B$, $\forall i \in \{1, 2, ..., n\}$. $\forall j \in \{1, 2, ..., n\}$.

**Fase de Recuperación.** La fase de recuperación de las memorias autoasociativas $\alpha\beta$ tipo Λ tiene dos casos posibles. En el primer caso el patrón de entrada es un patrón fundamental; es decir, la entrada es un patrón $\mathbf{x}^{\omega}$, con $\omega \in \{1, 2, ..., p\}$. En el segundo caso, el patrón de entrada NO es un patrón fundamental, sino la versión distorsionada de por lo menos uno de los patrones fundamentales; lo anterior significa que si el patrón de entrada es $\tilde{\mathbf{x}}$, debe existir al menos un valor de índice $\omega \in \{1, 2, ..., p\}$, que corresponde al patrón fundamental respecto del cual $\tilde{\mathbf{x}}$ es una versión alterada de alguno de los tres tipos: aditivo, sustractivo o mezclado.

**CASO 1: Patrón fundamental.** Se presenta a un patrón $\mathbf{x}^{\omega}$, con $\omega \in \{1, 2, ..., p\}$ a la memoria autoasociativa $\alpha\beta$ tipo V y se realiza la operación $\nabla_{\beta}$:

$$\boldsymbol{\Lambda}\, \Delta_{\beta}\, \mathbf{x}^{\omega}$$

El resultado de la operación anterior será el vector columna de dimensión *n*.

$$\left(\boldsymbol{\Lambda} \nabla_{\beta} \mathbf{x}^{\omega}\right)_i = \bigvee_{j=1}^{n} \beta(\lambda_{ij}, x_j^{\omega})$$

$$\left(\Lambda \nabla_{\beta} \mathbf{x}^{\omega}\right)_i = \bigvee_{j=1}^{n} \beta\left\{\left[\bigwedge_{\mu=1}^{p} \alpha(x_i^{\mu}, x_j^{\mu})\right], x_j^{\omega}\right\}$$

**CASO 2: Patrón alterado.** Se presenta el patrón binario $\tilde{\mathbf{x}}$ (patrón alterado de algún patrón fundamental $\mathbf{x}^{\omega}$) que es un vector columna de dimensión *n*, a la memoria autoasociativa $\alpha\beta$ tipo $\Lambda$ y se realiza la operación:

$$\Lambda\nabla_{\beta}\tilde{\mathbf{x}}$$

Al igual que en el caso 1, el resultado de la operación anterior es un vector columna de dimensión *n*, cuya *i*-ésima componente se expresa de la siguiente manera:

$$\left(\Lambda\nabla_{\beta}\tilde{\mathbf{x}}\right)_i = \bigvee_{j=1}^{n} \beta(\lambda_{ij}, \tilde{x}_j)$$

$$\left(\Lambda\nabla_{\beta}\tilde{\mathbf{x}}\right)_i = \bigvee_{j=1}^{n} \beta\left\{\left[\bigwedge_{\mu=1}^{p} \alpha(x_i^{\mu}, x_j^{\mu})\right], \tilde{x}_j\right\}$$

**Lema 4.31.** *Una memoria autoasociativa Alfa-Beta tipo $\Lambda$ tiene únicamente unos en la diagonal principal.*

**Demostración**. La *ij*-ésima entrada de una memoria autoasociativa Alfa-Beta tipo $\boldsymbol{\Lambda}$ está dada por $\lambda_{ij} = \bigwedge_{\mu=1}^{p} \alpha\left(x_i^{\mu}, x_j^{\mu}\right)$. Las entradas de la diagonal principal se obtienen de la expresión anterior haciendo $i = j$:

$$\lambda_{ii} = \bigwedge_{\mu=1}^{p} \alpha\left(x_i^{\mu}, x_i^{\mu}\right), \quad \forall i \in \{1,2,...n\}$$

Por la propiedad de la tabla se tiene que $\alpha(x_i^{\mu}, x_i^{\mu}) = 1$, por lo que la expresión anterior se transforma en:

$$\lambda_{ii} = \bigwedge_{\mu=1}^{p} (1) = 1, \quad \forall i \in \{1,2,...n\}$$

**Teorema 4.32.** Una memoria autoasociativa Alfa-Beta tipo $\boldsymbol{\Lambda}$ recupera de manera perfecta el conjunto fundamental completo; además, tiene máxima capacidad de aprendizaje.

**Demostración.-** Sea $\omega = \{1, 2, ..., p\}$ arbitrario. De acuerdo con el lema 4.31, para cada $i \in \{1, ..., n\}$ escogida arbitrariamente $\exists\, j_0 = i \in \{1, ..., n\}$ que cumple con:

$$\lambda_{ii} = 1 = \alpha(x_i^{\omega}, x_i^{\omega})$$

Es decir, para $i \in \{1, ..., n\}$ escogida arbitrariamente, $\exists\, j_0 = i \in \{1, ..., n\}$ que cumple con:

$$\lambda_{ij_0} = \alpha(x_i^{\omega}, x_{j_0}^{\omega})$$

Por lo tanto, de acuerdo con el Teorema 4.20

$$\mathbf{\Lambda\nabla}_\beta \mathbf{x}^\omega = \mathbf{x}^\omega, \forall \omega \in \{1, 2, ..., p\}$$

Esto significa que la memoria autoasociativa Alfa-Beta tipo Λ recupera de manera perfecta el conjunto fundamental completo.

Además, en la demostración de este Teorema, en ningún momento aparece restricción alguna sobre *p* que es la cardinalidad del conjunto fundamental; y esto quiere decir que el conjunto fundamental puede crecer tanto como se quiera. La consecuencia directa es que el número de patrones que puede aprender una memoria autoasociativa Alfa-Beta tipo Λ, con recuperación perfecta, es máximo.

El Teorema 4.32 se puede enunciar desde un enfoque matricial de la siguiente manera: dado que para cada asociación $(\mathbf{x}^\omega, \mathbf{x}^\omega)$ del conjunto fundamental de una memoria autoasociativa Alfa-Beta Λ, se cumple que cada fila de la matriz $\mathbf{x}^\omega \otimes (\mathbf{x}^\omega)^t$ - Λ contiene una entrada cero, entonces de la memoria Λ recupera el conjunto completo de patrones fundamentales en forma perfecta.

**Teorema 4.33** Sea $\{(\mathbf{x}^\mu,\mathbf{x}^\mu) \mid \mu = 1, 2, ..., p\}$ el conjunto fundamental de una memoria autoasociativa Alfa-Beta representada por Λ, y sea $\tilde{\mathbf{x}} \in A^n$ un patrón alterado con ruido sustractivo respecto a algún patrón fundamental $\mathbf{x}^\omega$ con $\omega \in \{1, 2, ..., p\}$. Si se presenta $\tilde{\mathbf{x}}$ a la memoria Λ como entrada, y si además para cada $i \in \{1, ..., n\}$ se cumple la condición de que $\exists j = j_0 \in \{1, ..., n\}$, el cual depende de $\omega$ y de $i$ tal que $\lambda_{ij_0} \leq \alpha(x^\omega, \tilde{x}_{j_0})$, entonces la recuperación $\Lambda\nabla_\beta \tilde{\mathbf{x}}$ es perfecta, es decir, $\Lambda\nabla_\beta \tilde{\mathbf{x}} = \mathbf{x}^\omega$

Demostración.- Por hipótesis se tiene que $\mathbf{y}^\mu = \mathbf{x}^\mu \forall \mu \in \{1, 2, ..., p\}$ y, por consiguiente, *m* = *n*. Al establecer estas dos condiciones en el Teorema 4.23 (Ver referencia), se obtiene el resultado: $\Lambda\nabla_\beta \tilde{\mathbf{x}} = \mathbf{x}^\omega$

El Teorema 4.33 nos dice que las memorias autoasociativas Alfa-Beta tipo V son inmunes a cierta cantidad de ruido sustractivo.

# CAPÍTULO 4

# Modelo Propuesto

En este capítulo se describe el nuevo modelo original de memoria asociativa bidireccional que se propone como tema central de este trabajo de tesis: las Memorias Asociativas Bidireccionales Alfa-Beta. En la sección 4.1 se describen de manera general las etapas funcionales del nuevo modelo, relacionándolas con el algoritmo que se presenta completo en la sección 4.4, junto con un ejemplo que ilustra el diseño y funcionamiento del nuevo modelo de BAM propuesto. Los pasos de este algoritmo se basan en los fundamentos teóricos presentados en la sección 4.2 para las etapas 1 y 3, y en la sección 4.3 para las etapas 2 y 4; en ambas secciones se desarrolla detalladamente el fundamento matemático original que sustenta el diseño, operación y aplicaciones del nuevo modelo.
Se incluye, por último, la sección 4.5 donde se realiza el análisis de complejidad del algoritmo, tanto en espacio como en tiempo.

## 4.1 Descripción de las Memorias Asociativas Bidireccionales Alfa-Beta

En general, cualquier modelo de memoria asociativa bidireccional presente en la literatura científica actual se podría esquematizar como se muestra en la figura 4.1.


**Figura 4.1** Esquema general de una Memoria Asociativa Bidireccional.

La BAM general es una "caja negra" que opera de la siguiente forma: dado un patrón **x** obtiene el patrón asociado **y**, y dado el patrón **y** obtiene el patrón asociado **x**. Además, si se asume que $\tilde{\mathbf{x}}$ y $\tilde{\mathbf{y}}$ son las versiones ruidosas de **x** y **y**, respectivamente, se espera que la BAM recupere los patrones correspondientes, **x** y **y**, libres de ruido.

Por ejemplo, el modelo de Kosko [14] podría esquematizarse como se muestra en la figura 4.2.

El modelo propuesto en este trabajo de tesis deberá comportarse de la misma forma que la BAM general en lo que respecta a la operación al recuperar patrones. El modelo se ha denominado Memorias Asociativas Bidireccionales Alfa-Beta (*Alpha-Beta BAM*) porque las memorias asociativas Alfa-Beta, tanto *max* como *min*, juegan un papel central en el diseño de este nuevo modelo.

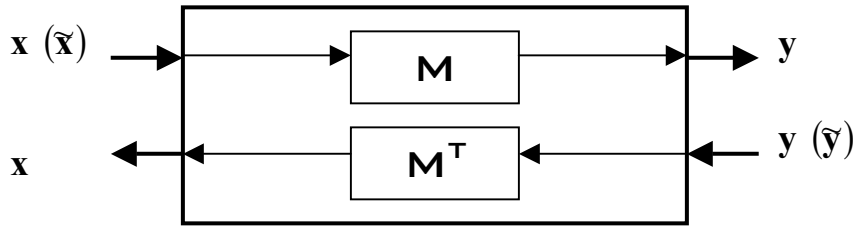



**Figura 4.2** Modelo de Kosko según el esquema de la BAM general.

Dado que es una memoria asociativa bidireccional, la recuperación de patrones debe darse en dos direcciones opuestas. En cada una de las dos direcciones, el modelo consta de dos etapas, y las cuatro etapas se muestran en la figura 4.3.

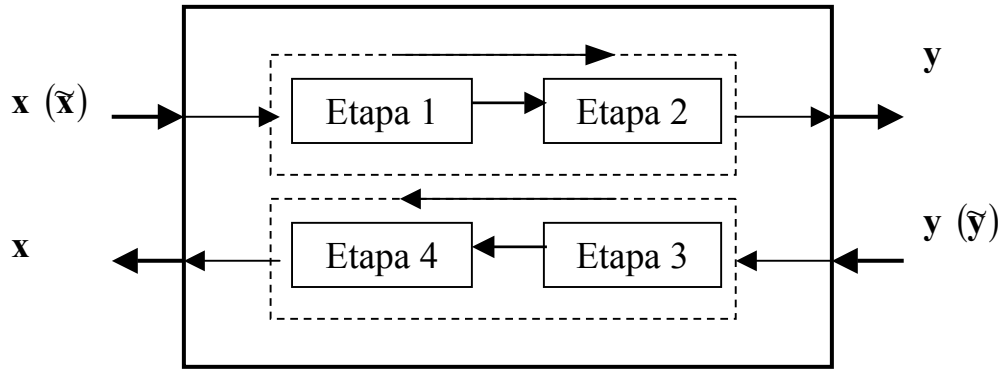



**Figura 4.3** Esquema de las etapas de una memoria asociativa bidireccional Alfa-Beta.

En esta tesis se asumirá que las memorias asociativas Alfa-Beta tienen un conjunto fundamental denotado por $\{(\mathbf{x}^\mu, \mathbf{y}^\mu) \mid \mu = 1, 2, ..., p\}$ $\mathbf{x}^\mu \in A^n$ y $\mathbf{y}^\mu \in A^m$, con $A = \{0, 1\}$, $n \in \mathbf{Z}^+$, $p \in \mathbf{Z}^+$, $m \in \mathbf{Z}^+$ y $1 < p \leq \min(2^n, 2^m)$. Además, se cumple la condición de que todos los patrones de entrada sean diferentes; es decir $\mathbf{x}^\mu = \mathbf{x}^\xi$ si y sólo si $\mu = \xi$. Si $\forall \mu \in \{1, 2, ..., p\}$ se cumple que $\mathbf{x}^\mu = \mathbf{y}^\mu$, la memoria Alfa-Beta será *autoasociativa*; y si la afirmación es falsa, es decir $\exists \mu \in \{1, 2, ..., p\}$ para el que se cumple $\mathbf{x}^\mu \neq \mathbf{y}^\mu$, la memoria Alfa-Beta será *heteroasociativa*.

Antes de continuar con la descripción del proceso y el desarrollo de los fundamentos matemáticos, se definen tres tipos de vectores que se usarán intensivamente en el nuevo modelo y se definen, además, dos transformadas vectoriales originales, propias del nuevo modelo.

Definición 1 (One-Hot) *Sea el conjunto* $A = \{0, 1\}$ *y sean* $p \in \mathbf{Z}^+$, $p > 1$, $k \in \mathbf{Z}^+$, *tales que* $1 \leq k \leq p$. *El k-ésimo vector one-hot de p bits se define como el vector* $\mathbf{h}^k \in A^p$ *para el cual se cumple que la k-ésima componente* $h_k^k = 1$ *y las demás componentes* $h_j^k = 0$, $\forall j \neq k$, $1 \leq j \leq p$.

Nota *En esta definición se excluye el valor* $p = 1$ *porque un vector one-hot de dimensión* 1, *por su esencia misma, no tiene razón de ser.*

Definición 2 (Zero-Hot) *Sea el conjunto* $A = \{0, 1\}$ *y sean* $p \in \mathbf{Z}^+$, $p > 1$, $k \in \mathbf{Z}^+$, *tales que* $1 \leq k \leq p$. *El k-ésimo vector zero-hot de p bits se define como el vector* $\bar{\mathbf{h}}^k \in A^p$ *para el cual se cumple que la k-ésima componente* $\bar{h}_k^k = 0$ *y las demás componentes* $\bar{h}_j^k = 1$, $\forall j \neq k, 1 \leq j \leq p$.

Nota *En esta definición se excluye el valor* $p = 1$ *porque un vector zero-hot de dimensión* 1, *por su esencia misma, no tiene razón de ser.*

Definición 3 (Transformada vectorial de expansión dimensional) *Sea el conjunto* $A = \{0, 1\}$ *y sean* $n \in \mathbf{Z}^+$, $m \in \mathbf{Z}^+$. *Dados dos vectores cualesquiera* $\mathbf{x} \in A^n$ *y* $\mathbf{e} \in A^m$, *se define la transformada vectorial de expansión de orden m,* $\tau^e : A^n \rightarrow A^{n+m}$, *como* $\tau^e(\mathbf{x}, \mathbf{e}) = \mathbf{X} \in A^{n+m}$, *vector cuyas componentes son:* $X_i = x_i$ *para* $1 \leq i \leq n$ *y* $X_i = e_i$ *para* $n + 1 \leq i \leq n + m$.

Definición 4 (Transformada vectorial de contracción dimensional) *Sea el conjunto* $A = \{0, 1\}$ *y sean* $n \in \mathbf{Z}^+$, $m \in \mathbf{Z}^+$ *tales que* $1 \leq m < n$. *Dado un vector cualesquiera* $\mathbf{X} \in A^{n+m}$, *se define la transformada vectorial de contracción de orden m,* $\tau^c : A^{n+m} \rightarrow A^m$, *como* $\tau^c(\mathbf{X}, m) = \mathbf{c} \in A^m$, *vector cuyas componentes son:* $c_i = X_{i+n}$ *para* $1 \leq i < m$.

Definición 5 (Vector negado) *Sea el conjunto* $A = \{0, 1\}$ *y sea un vector* $\mathbf{s} \in A^n$, *se define el vector negado de* $\mathbf{s}$ *como el vector* $\bar{\mathbf{s}}$, *tal que* $\bar{s}_i = \neg s_i$, *donde* $\neg$ *es el operador lógico de negación booleano.*

Habiendo definido cinco conceptos importantes, se continúa con el proceso de la descripción del funcionamiento de las memorias asociativas bidireccionales Alfa-Beta.

En la dirección $\mathbf{x} \rightarrow \mathbf{y}$, las etapas 1 y 2 tienen como función proporcionar un $\mathbf{y}^k$ a la salida ($k = 1, ..., p$) dado un $\mathbf{x}^k$ a la entrada, asumiendo que en el conjunto fundamental se incluye la asociación ($\mathbf{x}^k$, $\mathbf{y}^k$), como se ilustra en la figura 4.4.

El modelo está diseñado de tal forma que la Etapa 2, constituida principalmente por el *Linear Associator* modificado, entregue como salida el vector $\mathbf{y}^k$; si recordamos que el *Linear Associator* tiene recuperación correcta cuando a la entrada se presentan vectores ortonormales, es deseable que a la salida de la Etapa 1 se tengan precisamente vectores de este tipo. En efecto, además de la Etapa 1 que es la contribución principal de esta tesis, otra contribución importante es la Etapa 2: en esta tesis se presentará una variación original del *Linear Associator* que permite obtener $\mathbf{y}^k$ a partir de un vector $\mathbf{h}^k$ *one-hot* en su *k*-ésima coordenada.

Por tanto, ya se tiene resuelta la tarea que debe realizar la Etapa 2. De la figura 4.4 se puede observar que falta descubrir qué hace la Etapa 1.

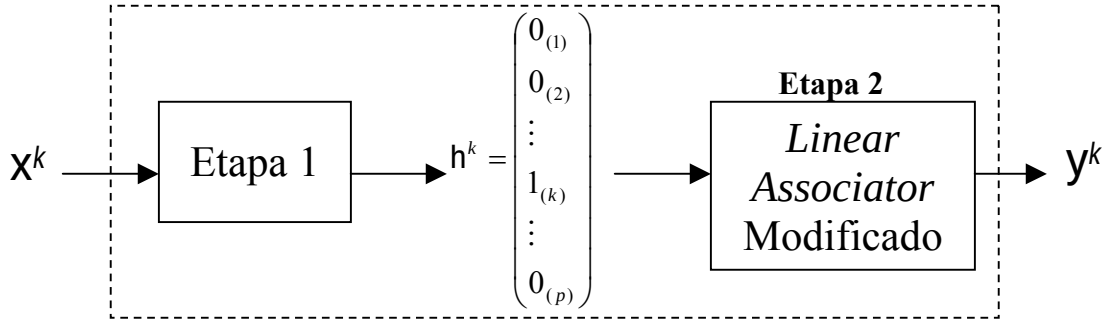



**Figura 4.4** Esquema del proceso a realizar en el sentido $\mathbf{x} \rightarrow \mathbf{y}$.

La tarea de la Etapa 1 es: dado un $\mathbf{x}^k$ o una versión ruidosa de éste ($\tilde{\mathbf{x}}^k$), se debe obtener sin ambigüedad y sin ninguna condición adicional, el vector *one-hot* $\mathbf{h}^k$.

La Etapa 1 juega un papel importante en la fase de aprendizaje, y realiza lo siguiente:

Como primer paso, se toma el conjunto fundamental $\{(\mathbf{x}^\mu, \mathbf{x}^\mu) \mid \mu = 1, ..., p\}$, al que llamaremos Conjunto Fundamental Original de $\mathbf{x}$ ($CFO_x$). A continuación, a cada uno de los patrones $\mathbf{x}^\mu$ se le aplica la transformada vectorial de expansión dimensional del vector $\mathbf{x}^\mu$ con su correspondiente vector *one-hot* de $p$ bits, esto es:

$$\tau^e (\mathbf{x}^\mu, \mathbf{h}^\mu) = \mathbf{X}^\mu$$

Ahora tenemos el nuevo conjunto fundamental $\{(\mathbf{X}^\mu, \mathbf{X}^\mu) \mid \mu = 1, ..., p\}$, al que denominaremos Conjunto Fundamental *one-hot* de $\mathbf{X}$ ($CFoh_X$). Con el $CFoh_X$ se crea la memoria autoasociativa Alfa-Beta *max*, de la siguiente manera:

$$\mathbf{V}_x = \bigvee_{\mu=1}^{p} \left[ \mathbf{X}^\mu \otimes (\mathbf{X}^\mu)^t \right]$$

De nuevo se toma el $CFO_x$. A cada uno de los patrones del $CFO_x$ se le aplica la transformada vectorial de expansión dimensional del vector $\mathbf{x}^\mu$ con su correspondiente vector *zero-hot* de $p$ bits, y obtenemos:

$$\tau^e (\mathbf{x}^\mu, \overline{\mathbf{h}}^\mu) = \overline{\mathbf{X}}^\mu$$

Se obtiene el nuevo conjunto fundamental $\{(\overline{\mathbf{X}}^\mu, \overline{\mathbf{X}}^\mu) \mid \mu = 1, ..., p\}$, denominado conjunto fundamental *zero-hot* de $\mathbf{X}$ ($CFzh_X$). Con el $CFzh_X$ se crea la memoria autoasociativa Alfa-Beta *min*, de la siguiente manera:

$$\mathbf{\Lambda}_x = \bigwedge_{\mu=1}^{p} \left[ \overline{\mathbf{X}}^\mu \otimes (\overline{\mathbf{X}}^\mu)^t \right]$$

Al presentar como entrada en la Etapa 1 el vector $\mathbf{x}^k$, y al operar las memorias $\mathbf{V}_x$ y $\mathbf{\Lambda}_x$ obtenidas en los pasos anteriores, se obtiene como resultado, de acuerdo con el algoritmo detallado en la sección 4.3, el vector *one-hot* $\mathbf{h}^k$ que será la entrada de la Etapa 2.

En la Etapa 2 se construye un *Linear Associator* con los patrones $\mathbf{y}^\mu$, usando un algoritmo de modificación original de este trabajo de tesis. Esta modificación consiste en aprovechar las características del conjunto de vectores *one-hot* para así obviar la fase de aprendizaje del *Linear Associator*, la cual se realiza implícitamente: se toman los vectores $\mathbf{y}^\mu$ con $\mu = 1, 2, ..., p$ y se acomodan en una matriz **LAy** de la siguiente manera:

$$\mathbf{LAy} = \begin{bmatrix} y_1^1 & y_1^2 & \cdots & y_1^p \\ y_2^1 & y_2^2 & \cdots & y_2^p \\ \vdots & \vdots & \cdots & \vdots \\ y_m^1 & y_m^2 & \cdots & y_m^p \end{bmatrix}$$

Así, en la fase de recuperación, se presenta a la entrada de **LAy** un patrón *one-hot*, digamos $\mathbf{h}^k$, y se obtiene de inmediato el correspondiente $\mathbf{y}^k$ a la salida, que también es la salida de la memoria asociativa bidireccional Alfa-Beta en el sentido $\mathbf{x} \rightarrow \mathbf{y}$.

Sólo resta describir el proceso que se sigue en el sentido de $\mathbf{y} \rightarrow \mathbf{x}$ para evidenciar la bidireccionalidad del modelo. Este proceso se esquematiza en la figura 4.5.

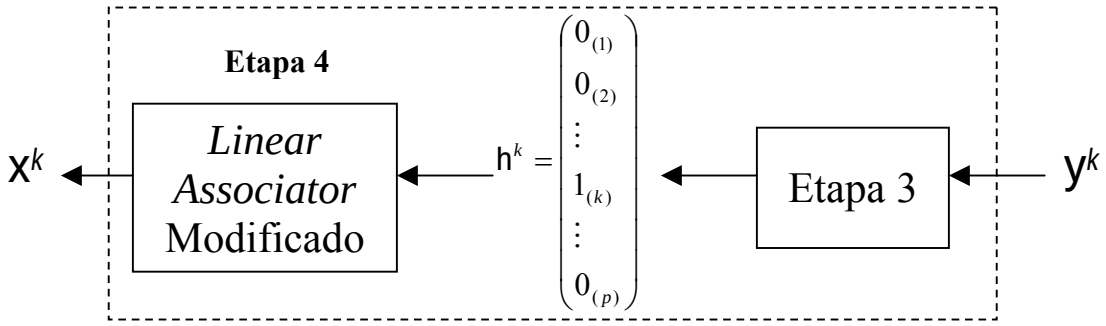



**Figura 4.5** Esquema del proceso realizado en el sentido de $\mathbf{y} \rightarrow \mathbf{x}$.

La tarea de la Etapa 3 es similar a la tarea de la Etapa 1, pero con argumentos $\mathbf{y}^k$: dado un $\mathbf{y}^k$ o una versión ruidosa de éste ($\tilde{\mathbf{y}}^k$), se debe obtener sin ambigüedad y sin ninguna condición adicional, el vector *one-hot* $\mathbf{h}^k$.

Como primer paso, se toma como conjunto fundamental $\{(\mathbf{y}^\mu, \mathbf{y}^\mu) \mid \mu = 1, ..., p\}$, al que llamaremos Conjunto Fundamental Original de **y** ($\text{CFO}_y$). A continuación, cada uno de los patrones $\mathbf{y}^\mu$ se le aplica la transformada vectorial de expansión dimensional del vector $\mathbf{y}^\mu$ con su correspondiente vector *one-hot* de $p$ bits, esto es:

$$\tau^{e}(\mathbf{y}^{\mu}, \mathbf{h}^{\mu}) = \mathbf{Y}^{\mu}$$

Ahora tenemos el nuevo conjunto fundamental $\{(\mathbf{Y}^{\mu}, \mathbf{Y}^{\mu}) \mid \mu = 1, ..., p\}$, que denominaremos Conjunto Fundamental *one-hot* de $\mathbf{Y}$ ($\text{CFoh}_Y$). Con el $\text{CFoh}_Y$ se crea la memoria autoasociativa Alfa-Beta *max*, de la siguiente manera:

$$\mathbf{V}_y = \bigvee_{\mu=1}^{p} \left[ \mathbf{Y}^{\mu} \otimes (\mathbf{Y}^{\mu})^t \right]$$

De nuevo, se toma el $\text{CFO}_y$. A cada uno los patrones del $\text{CFO}_y$ se le aplica la transformada vectorial de expansión dimensional del vector $\mathbf{y}^{\mu}$ con su correspondiente vector *zero-hot* de $p$ bits, lo que resulta en:

$$\tau^{e}(\mathbf{y}^{\mu}, \overline{\mathbf{h}}^{\mu}) = \overline{\mathbf{Y}}^{\mu}$$

Se obtiene el nuevo conjunto fundamental $\{(\overline{\mathbf{Y}}^{\mu}, \overline{\mathbf{Y}}^{\mu}) \mid \mu = 1, ..., p\}$, denominado conjunto fundamental *zero-hot* de $\mathbf{Y}$ ($\text{CFzh}_Y$). Con el $\text{CFzh}_Y$ se crea la memoria autoasociativa Alfa-Beta *min*, de la siguiente manera:

$$\mathbf{\Lambda}_y = \bigwedge_{\mu=1}^{p} \left[ \overline{\mathbf{Y}}^{\mu} \otimes (\overline{\mathbf{Y}}^{\mu})^t \right]$$

De manera similar a la Etapa 2, en la Etapa 4 se construye un *Linear Associator* modificado con los patrones $\mathbf{x}^{\mu}$ con $\mu = 1, 2, ..., p$ y se acomodan en una matriz **LAx** de la siguiente manera:

$$\mathbf{LAx} = \begin{bmatrix} x_1^1 & x_1^2 & \cdots & x_1^p \\ x_2^1 & x_2^2 & \cdots & x_2^p \\ \vdots & \vdots & \cdots & \vdots \\ x_n^1 & x_n^2 & \cdots & x_n^p \end{bmatrix}$$

Hasta este momento, se describió el proceso para la construcción y operación de las memorias asociativas bidireccionales Alfa-Beta; es decir, se han descrito los pasos necesarios para concretar ambas fases: la de aprendizaje y la de recuperación de patrones.

En las siguientes dos secciones se presentará detalladamente el fundamento teórico que sustenta el funcionamiento de las memorias asociativas bidireccionales Alfa-Beta.

## 4.2 Fundamento Teórico de las Etapas 1 y 3

El algoritmo de la Etapa 1 y, por consiguiente, de la etapa 3 es la contribución más importante de este trabajo de tesis.

A continuación se enuncian 5 Teoremas y 9 Lemas con su demostración correspondiente. Este fundamento matemático es la sustentación de los pasos requeridos por el algoritmo completo el cual se presenta en la sección 4.4.

Por convención de utilizará el símbolo ■ para indicar el final de una demostración.

Teorema 4.1 *Sea* $\{(\mathbf{x}^{\mu}, \mathbf{x}^{\mu}) \mid \mu = 1,2,..., p\}$ *el conjunto fundamental de una memoria autoasociativa Alfa-Beta max representada por* $\mathbf{V}$*, y sea* $\tilde{\mathbf{x}} \in A^n$ *un patrón alterado con ruido aditivo respecto de algún patrón fundamental* $\mathbf{x}^{\omega}$ *con* $\omega \in \{1, 2, ..., p\}$*. Asumamos que en la fase de recuperación se presenta* $\tilde{\mathbf{x}}$ *a la memoria* $\mathbf{V}$ *como entrada y consideremos un índice* $k \in \{1, ..., n\}$*. La k-ésima componente recuperada* $\left(\mathbf{V}\Delta_{\beta}\tilde{\mathbf{x}}\right)_k$ *es precisamente* $x_k^{\omega}$ *si y sólo si se cumple la condición de que* $\exists r \in \{1, .., n\}$*, el cual depende de* $\omega$ *y de k tal que* $v_{kr} \leq \alpha(x_k^{\omega}, \tilde{x}_r)$.

Demostración.-

$\Rightarrow$) Por hipótesis se asume que $\left(\mathbf{V}\Delta_{\beta}\tilde{\mathbf{x}}\right)_k = x_k^{\omega}$. Por contradicción, ahora supongamos que es falsa la afirmación de que $\exists r \in \{1, .., n\}$ tal que $v_{kr} \leq \alpha(x_k^{\omega}, \tilde{x}_r)$; esto es equivalente a afirmar que $\forall\ r \in \{1, .., n\}$ $v_{kr} > \alpha(x_k^{\omega}, \tilde{x}_r)$, lo cual equivale a que $\forall\ r \in \{1, .., n\}$ $\beta(v_{kr}, \tilde{x}_r) > \beta\left[\alpha(x_k^{\omega}, \tilde{x}_r), \tilde{x}_r\right] = x_k^{\omega}$. Al tomar mínimos en ambos lados de la desigualdad respecto del índice *r*, se tiene que

$$\bigwedge_{r=1}^{n} \beta(v_{kr}, \tilde{x}_r) > \bigwedge_{r=1}^{n} x_k^{\omega} = x_k^{\omega}$$

y esto significa que $\left(\mathbf{V}\Delta_{\beta}\tilde{\mathbf{x}}\right)_k = \bigwedge_{r=1}^{n} \beta(v_{kr}, \tilde{x}_r) > x_k^{\omega}$, afirmación que contradice la hipótesis.

$\Leftarrow$) Al cumplirse las condiciones del Teorema 4.30 [15] para cada $i \in \{1, ..., n\}$, se tiene que $\mathbf{V}\Delta_{\beta}\tilde{\mathbf{x}} = \mathbf{x}^{\omega}$; es decir se cumple que $\left(\mathbf{V}\Delta_{\beta}\tilde{\mathbf{x}}\right)_i = x_i^{\omega}$, $\forall\ i \in \{1, ..., n\}$. Al fijar los índices *i* y $j_0$ de modo que $i = k$ y $j_0 = r$ (que depende de $\omega$ y de *k*) se obtiene el resultado deseado $\left(\mathbf{V}\Delta_{\beta}\tilde{\mathbf{x}}\right)_k = x_k^{\omega}$.■

Ejemplo 4.1 (Ilustra el Teorema 4.1) Sean $p = 4$ y $n = 4$. En conjunto fundamental para una memoria autoasociativa contiene cuatro parejas de patrones $\{(\mathrm{x}^{\mu}, \mathrm{x}^{\mu}) \mid \mu = 1, 2, 3, 4\}$. Cada vector $\mathrm{x}^{\mu}$ es un vector columna con valores en el conjunto $A^4$ y los valores de las componentes para cada vector son los siguientes:

$$\mathbf{x}^1 = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}, \quad \mathbf{x}^2 = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}, \quad \mathbf{x}^3 = \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \end{bmatrix}, \quad \mathbf{x}^4 = \begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \end{bmatrix}$$

La matriz de la memoria autoasociativa Alfa-Beta *max* para este conjunto fundamental es:

$$\mathbf{V} = \begin{bmatrix} 1 & 1 & 2 & 2 \\ 1 & 1 & 2 & 2 \\ 2 & 2 & 1 & 2 \\ 1 & 1 & 1 & 1 \end{bmatrix}$$

Ahora supongamos que le presentamos a la matriz **V** una versión ruidosa $\tilde{\mathbf{x}}$ del vector $\mathbf{x}^{\omega}$ con $\omega = 2$, con ruido aditivo, cuyos valores de sus componentes son:

$$\tilde{\mathbf{x}} = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} \text{ entonces el vector recuperado será } \mathbf{V}\Delta_{\beta}\tilde{\mathbf{x}} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

Se puede observar que se recuperó de manera correcta el vector $\mathbf{x}^2$; sin embargo, lo que nos interesa es comprobar la recuperación de componentes y que, además, se cumple la condición del teorema 4.1.

La primera componente recuperada por la memoria autoasociativa Alfa-Beta *max* es igual a cero y que es precisamente el valor que tiene la primera componente del segundo patrón, esto es, $\left(\mathbf{V}\Delta_{\beta}\tilde{\mathbf{x}}\right)_1 = 0 = x_1^2$. Ahora veamos si se cumple la condición de que $\exists r \in \{1, .., n\}$, tal que $\nu_{kr} \leq \alpha(x_k^{\omega}, \tilde{x}_r)$. Para nuestro ejemplo $k = 1$ y $\omega = 2$.

Para $r = 1$, $\nu_{11} = 1$ y $\alpha(x_1^2, \tilde{x}_1) = \alpha(0,1) = 0$, esto es $\nu_{11} > \alpha(x_1^2, \tilde{x}_1)$, no cumple
Para $r = 2$, $\nu_{12} = 1$ y $\alpha(x_1^2, \tilde{x}_2) = \alpha(0,0) = 1$, esto es $\nu_{12} \leq \alpha(x_1^2, \tilde{x}_2)$, si cumple
Para $r = 3$, $\nu_{13} = 2$ y $\alpha(x_1^2, \tilde{x}_3) = \alpha(0,0) = 1$, esto es $\nu_{13} > \alpha(x_1^2, \tilde{x}_3)$, no cumple
Para $r = 4$, $\nu_{14} = 2$ y $\alpha(x_1^2, \tilde{x}_4) = \alpha(0,0) = 1$, esto es $\nu_{14} > \alpha(x_1^2, \tilde{x}_4)$, no cumple

Por lo tanto, existe $r = 2$ tal que $\nu_{12} \leq \alpha(x_1^2, \tilde{x}_2)$.

Lema 4.1 *Sea* $\{(\mathbf{X}^k, \mathbf{X}^k) \mid k = 1, ..., p\}$ *el conjunto fundamental de una memoria autoasociativa Alfa-Beta max representada por* **V**, *con* $\mathbf{X}^k = \tau^e(\mathbf{x}^k, \mathbf{h}^k)$ *para* $k = 1, ..., p$, *y sea* $\mathbf{F} = \tau^e(\mathbf{x}^k, \mathbf{u}) \in A^{n+p}$ *una versión alterada con ruido aditivo de un patrón específico* $\mathbf{X}^k$, *siendo* $\mathbf{u} \in A^p$ *el vector definido como* $\mathbf{u} = \sum_{i=1}^{p} \mathbf{h}^i$. *Si en la fase de recuperación se presenta* **F** *a la entrada de la memoria* **V**, *entonces la componente* $X^k_{n+k}$ *se recupera de manera correcta; es decir,* $\left(\mathbf{V}\Delta_{\beta}\mathbf{F}\right)_{n+k} = X^k_{n+k} = 1$ .

Demostración.- Esta demostración se hará para dos casos mutuamente exclusivos.

CASO 1 El patrón **F** contiene una componente con valor 0. Esto significa que $\exists\, j \in \{1, ..., n+p\}$ tal que $F_j = 0$; además, por la forma de construir el vector $\mathbf{X}^k$ es claro que $X^k_{n+k} = 1$ . Por ello $\alpha(X^k_{n+k}, F_j) = \alpha(1,0) = 2$ y, dado que el máximo valor posible para alguna componente de la memoria **V** es 2 se tiene que $v_{(n+k)j} \le \alpha(X^k_{n+k}, F_j)$ . De acuerdo con el Teorema 4.1, $X^k_{n+k}$ se recupera de manera correcta.

CASO 2 El patrón **F** no contiene una componente con valor 0; es decir $F_j = 1$ $\forall\, j \in \{1, ..., n+p\}$. Esto significa que no es posible garantizar la existencia de un valor $j \in \{1, ..., n+p\}$ tal que $v_{(n+k)j} \le \alpha(X^k_{n+k}, F_j)$ , y por ello no se puede aplicar el Teorema 4.1. Sin embargo, mostremos la imposibilidad de que $\left(\mathbf{V}\Delta_\beta \mathbf{F}\right)_{n+k} = 0$. La fase de recuperación de la memoria autoasociativa Alfa-Beta max **V** al tener al vector **F** a la entrada toma la siguiente forma para la $n + k$-ésima componente recuperada:

$$\left(\mathbf{V}\Delta_\beta \mathbf{F}\right)_{n+k} = \bigwedge_{j=1}^{n} \beta(v_{(n+k)j}, F_j) = \bigwedge_{j=1}^{n} \beta\left\{\left[\bigvee_{\mu=1}^{p} \alpha(X^\mu_{n+k}, X^\mu_j)\right], F_j\right\}$$

Por la forma de construir el vector $\mathbf{X}^k$, además de que $X^k_{n+k} = 1$, es importante notar que $X^\mu_{n+k} \neq 1, \quad \forall \mu \neq k$, y de aquí podemos establecer que

$$\bigvee_{\mu=1}^{p} \alpha(X^\mu_{n+k}, X^\mu_j) = \alpha(X^k_{n+k}, X^k_j) = \alpha(1, X^k_j)$$

es diferente de cero sin importar el valor de $X^k_j$. Al tomar en cuenta que $F_j = 1$ $\forall\, j \in \{1, ..., n+p\}$, podemos concluir que es imposible que

$$\left(\mathbf{V}\Delta_\beta \mathbf{F}\right)_{n+k} = \bigwedge_{j=1}^{n} \beta(\alpha(1, X^k_j), 1)$$

sea cero; es decir, $\left(\mathbf{V}\Delta_\beta \mathbf{F}\right)_{n+k} = 1 = X^k_{n+k}$ . ■

Ejemplo 4.2 (Ilustra el Caso 1 del Lema 4.1) Tomando el conjunto fundamental del ejemplo 4.1 se construyen los patrones $\mathbf{X}^k$ para $k = 1, 2, 3, 4$ utilizando la transformada vectorial de expansión de los vectores $\mathbf{x}^k$ de la Definición 3:

$$\mathbf{X}^1 = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}, \quad \mathbf{X}^2 = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \end{bmatrix}, \quad \mathbf{X}^3 = \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \end{bmatrix}, \quad \mathbf{X}^4 = \begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}$$

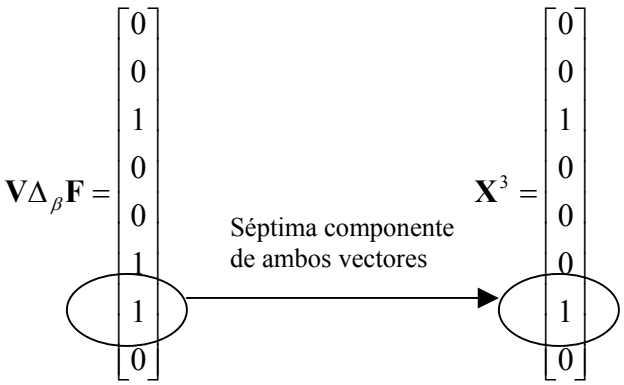
La matriz de la memoria autoasociativa Alfa-Beta *max* es:

$$\mathbf{V} = \begin{bmatrix} 1 & 1 & 2 & 2 & 2 & 2 & 2 & 1 \\ 1 & 1 & 2 & 2 & 2 & 2 & 2 & 2 \\ 2 & 2 & 1 & 2 & 2 & 2 & 2 & 2 \\ 1 & 1 & 1 & 1 & 1 & 2 & 2 & 2 \\ 1 & 1 & 1 & 1 & 1 & 2 & 2 & 2 \\ 2 & 2 & 2 & 2 & 2 & 1 & 2 & 2 \\ 2 & 2 & 1 & 2 & 2 & 2 & 1 & 2 \\ 1 & 1 & 2 & 2 & 2 & 2 & 2 & 1 \end{bmatrix}$$

Ahora, tomando $k = 3$ utilizaremos vector $\mathbf{X}^3$ y obtendremos su versión ruidosa **F**, con ruido aditivo:

$$\mathbf{X}^3 = \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \end{bmatrix}, \text{ el vector ruidoso, con ruido aditivo, } \mathbf{F} \text{ es: } \mathbf{F} = \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}$$

Al presentarle **F** a la matriz **V**, el vector recuperado es:

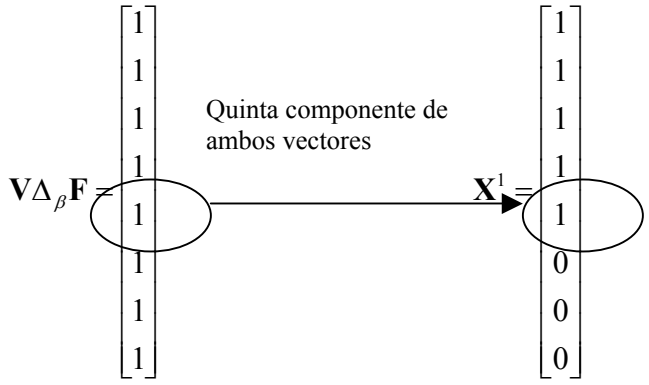



Recordemos que en este ejemplo $n = 4$ y $k = 3$, entonces $\left(\mathbf{V}\Delta_\beta\mathbf{F}\right)_{4+3} = X^3_{4+3} = 1$. Por lo tanto, se recupera de manera correcta la séptima componente del tercer patrón.

Ejemplo 4.3 (Ilustra el Caso 2 del Lema 4.1) Utilizando la matriz **V** obtenida en el ejemplo 4.2 y con $k = 1$ se obtiene el vector **F** que es la versión ruidosa, con ruido aditivo, de $\mathbf{X}^1$, cuyos valores de componentes son:

$$\mathbf{X}^1 = \begin{bmatrix}1\\1\\1\\1\\1\\0\\0\\0\end{bmatrix}, \text{ el vector ruidoso, con ruido aditivo, } \mathbf{F} \text{ es: } \mathbf{F} = \begin{bmatrix}1\\1\\1\\1\\1\\1\\1\\1\end{bmatrix}$$

Al presentarle **F** a la matriz **V**, el vector recuperado es:




Para este ejemplo $n = 4$ y $k = 1$, entonces $\left(\mathbf{V}\Delta_\beta\mathbf{F}\right)_{4+1} = X^1_{4+1} = 1$. Por lo tanto, se recupera de manera correcta la quinta componente del primer patrón.

Teorema 4.2 *Sea* $\{(\mathbf{X}^k, \mathbf{X}^k) \mid k = 1, ..., p\}$ *el conjunto fundamental de una memoria autoasociativa Alfa-Beta max representada por* $\mathbf{V}$*, con* $\mathbf{X}^k = \tau^e(\mathbf{x}^k, \mathbf{h}^k)$ *para* $k = 1, ..., p$*, y sea* $\mathbf{F} = \tau^e(\mathbf{x}^k, \mathbf{u}) \in A^{n+p}$ *una versión alterada con ruido aditivo de un patrón específico* $\mathbf{X}^k$*, siendo* $\mathbf{u} \in A^p$ *el vector definido como* $\mathbf{u} = \sum_{i=1}^{p} \mathbf{h}^i$*. Asumamos que en la fase de recuperación se presenta* $\mathbf{F}$ *a la entrada de la memoria* $\mathbf{V}$ *y se obtiene el patrón* $\mathbf{R} = \mathbf{V}\Delta_\beta\mathbf{F} \in A^{n+p}$*. Si al tomar como argumento el vector* $\mathbf{R}$ *se realiza la transformada vectorial de contracción* $\mathbf{r} = \tau^c(\mathbf{R}, n) \in A^p$*, entonces el vector* $\mathbf{r}$ *tiene dos posibilidades mutuamente exclusivas: o sucede que* $\exists\, k \in \{1, ..., p\}$ *tal que* $\mathbf{r} = \mathbf{h}^k$*, o* $\mathbf{r}$ *no es un vector one-hot.*

Demostración.- Por la definición de transformada vectorial de contracción se tiene que $r_i = R_{i+n} = (\mathbf{V}\Delta_\beta\mathbf{F})_{i+n}$ para $1 \le i \le p$, y en particular haciendo $i = k$ se tiene que $r_k = R_{k+n} = (\mathbf{V}\Delta_\beta\mathbf{F})_{k+n}$. Pero por el Lema 4.1 resulta que $\left(\mathbf{V}\Delta_\beta\mathbf{F}\right)_{n+k} = X^k_{n+k}$, y dado que $\mathbf{X}^k = \tau^e(\mathbf{x}^k, \mathbf{h}^k)$, el valor $X^k_{n+k}$ es igual al valor de la componente $h^k_k = 1$; es decir, $r_k = 1$. Al considerar que $r_k = 1$, el vector $\mathbf{r}$ tiene dos posibilidades mutuamente exclusivas: o sucede que $r_j = 0\ \forall\ j \ne k$ en cuyo caso $\mathbf{r} = \mathbf{h}^k$; o bien, ocurre que $\exists\, j \in \{1, ..., p\}, j \ne k$ para el cual $r_j = 1$, en cuyo caso no es posible que $\mathbf{r}$ sea un vector *one-hot*, atendiendo a la definición 1■

Ejemplo 4.4 (Ilustra el Teorema 4.2) Tomando la matriz $\mathbf{V}$ obtenida en el ejemplo 4.2 y con $k = 2$, los valores de las componentes $\mathbf{X}^2$ y su respectivo patrón ruidoso $\mathbf{F}$, con ruido aditivo, son

$$\mathbf{X}^2 = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \end{bmatrix} \quad \text{y} \quad \mathbf{F} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}$$

Al presentarle $\mathbf{F}$ a la matriz $\mathbf{V}$ se obtiene el vector $\mathbf{R}$:

$$\mathbf{R} = \mathbf{V}\Delta_{\beta}\mathbf{F} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \end{bmatrix}$$

Al tomar como argumento este vector y realizar la transformada vectorial de contracción del vector **R**, obtenemos el vector **r**,

$$\mathbf{r} = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \end{bmatrix}$$

y de acuerdo a la Definición 1, podemos observar que **r** es el segundo vector *one-hot* de 4 bits.

Ahora realicemos el mismo proceso pero para $k = 4$. Entonces, $\mathbf{X}^4$ y su respectivo patrón ruidoso **F**, con ruido aditivo, son:

$$\mathbf{X}^4 = \begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} \quad \text{y} \quad \mathbf{F} = \begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}$$

Al presentarle **F** a la matriz **V** se obtiene el vector **R**:

$$\mathbf{R} = \mathbf{V}\Delta_\beta \mathbf{F} = \begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 1 \end{bmatrix}$$

Al tomar como argumento este vector y realizar la transformada vectorial de contracción del vector **R** obtenemos el vector **r**,

$$\mathbf{r} = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 1 \end{bmatrix}$$

De acuerdo a la Definición 1 **r** no es un vector *one-hot*.

Teorema 4.3 *Sea* $\{(\mathbf{x}^\mu, \mathbf{x}^\mu) \mid \mu = 1, 2, ..., p\}$ *el conjunto fundamental de una memoria autoasociativa Alfa-Beta min representada por* $\mathbf{\Lambda}$*, y sea* $\tilde{\mathbf{x}} \in A^n$ *un patrón alterado con ruido sustractivo respecto de algún patrón fundamental* $x^\omega$ *con* $\omega \in \{1, 2, ..., p\}$*. Asumamos que en la fase de recuperación se presenta* $\tilde{\mathbf{x}}$ *a la memoria* $\mathbf{\Lambda}$ *como entrada y consideremos un índice* $k \in \{1, ..., n\}$*. La k-ésima componente recuperada* $\left(\mathbf{\Lambda}\nabla_\beta \tilde{\mathbf{x}}\right)_k$ *es precisamente* $x_k^\omega$ *si y sólo si se cumple la condición de que* $\exists\ r \in \{1, ..., n\}$*, el cual depende de* $\omega$ *y de k tal que* $\lambda_{kr} \geq \alpha\left(x_k^\omega, \tilde{x}_r\right)$.

Demostración.-

$\Rightarrow$) Por hipótesis se asume que $\left(\mathbf{\Lambda}\nabla_\beta \tilde{\mathbf{x}}\right)_k = x_k^\omega$. Por contradicción, ahora supongamos que es falsa la afirmación de que $\exists\ r \in \{1, ..., n\}$ tal que $\lambda_{kr} \geq \alpha\left(x_k^\omega, \tilde{x}_r\right)$; esto es equivalente a afirmar que $\forall\ r \in \{1, ..., n\}$ $\lambda_{kr} < \alpha\left(x_k^\omega, \tilde{x}_r\right)$, lo cual es equivalente a que $\forall\ r \in \{1, ..., n\}$ $\beta\left(\lambda_{kr}, \tilde{x}_r\right) < \beta\left[\alpha\left(x_k^\omega, \tilde{x}_r\right), \tilde{x}_r\right] = x_k^\omega$. Al tomar máximos en ambos lados de la desigualdad respecto del índice *r*, se tiene que

$$\bigvee_{r=1}^{n} \beta\left(\lambda_{kr}, \tilde{x}_r\right) < \bigvee_{r=1}^{n} x_k^\omega = x_k^\omega$$

y esto significa que $\left(\mathbf{\Lambda}\nabla_\beta \tilde{\mathbf{x}}\right)_k = \bigvee_{r=1}^{n} \beta\left(\lambda_{kr}, \tilde{x}_r\right) < x_k^\omega$, afirmación que contradice la hipótesis.

$\Leftarrow$) Al cumplirse las condiciones del Teorema 4.33 [15] para cada $i \in \{1, ..., n\}$, se tiene que $\mathbf{\Lambda}\nabla_\beta \tilde{\mathbf{x}} = \mathbf{x}^\omega$; es decir, se cumple que $\left(\mathbf{\Lambda}\nabla_\beta \tilde{\mathbf{x}}\right)_i = x_i^\omega \ \forall\ i \in \{1, ..., n\}$. Al fijar los índices $i$ y $j_0$ de modo que $i = k$ y $j_0 = r$ (que depende de $\omega$ y de $k$) se obtiene el resultado deseado $\left(\mathbf{\Lambda}\nabla_\beta \tilde{\mathbf{x}}\right)_k = x_k^\omega$ .■

Ejemplo 4.5 (Ilustra el Teorema 4.3) Los valores de las componentes del conjunto fundamental del ejemplo 4.1 se muestran a continuación con $p = 4$ y $n = 4$,

$$\mathbf{x}^1 = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}, \quad \mathbf{x}^2 = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}, \quad \mathbf{x}^3 = \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \end{bmatrix}, \quad \mathbf{x}^4 = \begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \end{bmatrix}$$

La matriz de la memoria autoasociativa Alfa-Beta *min* para este conjunto fundamental es:

$$\mathbf{\Lambda} = \begin{bmatrix} 1 & 1 & 0 & 1 \\ 1 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 \end{bmatrix}$$

Ahora supongamos que le presentamos a la matriz $\mathbf{\Lambda}$ una versión ruidosa $\tilde{\mathbf{x}}$ del vector $\mathbf{x}^\omega$ con $\omega = 4$, con ruido sustractivo, cuyos valores de sus componentes son:

$$\tilde{\mathbf{x}} = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} \text{ entonces el vector recuperado será } \mathbf{\Lambda}\nabla_\beta \tilde{\mathbf{x}} = \begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \end{bmatrix}$$

Se puede observar que se recuperó de manera correcta el vector $\mathbf{x}^4$; sin embargo, lo que nos interesa es comprobar la recuperación de componentes y que, además, se cumple la condición del teorema 4.5.

La segunda componente recuperada por la memoria autoasociativa Alfa-Beta *min* es igual a uno y es precisamente el valor que tiene la segunda componente del cuarto patrón, esto es, $\left(\mathbf{\Lambda}\nabla_\beta \tilde{\mathbf{x}}\right)_2 = 1 = x_2^4$. Ahora veamos si se cumple la condición de que $\exists r \in \{1, .., n\}$, tal que $\lambda_{kr} \geq \alpha(x_k^\omega, \tilde{x}_r)$. Para nuestro ejemplo $k = 2$ y $\omega = 4$.

Para $r = 1$, $\lambda_{21} = 1$ y $\alpha(x_2^4, \tilde{x}_1) = \alpha(1,1) = 1$, esto es $\lambda_{21} \geq \alpha(x_2^4, \tilde{x}_1)$, si cumple
Para $r = 2$, $\lambda_{22} = 1$ y $\alpha(x_2^4, \tilde{x}_2) = \alpha(1,0) = 2$, esto es $\lambda_{22} < \alpha(x_2^4, \tilde{x}_2)$, no cumple

Para $r = 3$, $\lambda_{23} = 0$ y $\alpha(x_2^4, \overline{x}_3) = \alpha(1,0) = 2$, esto es $\lambda_{23} < \alpha(x_2^4, \overline{x}_3)$, no cumple
Para $r = 4$, $\lambda_{24} = 1$ y $\alpha(x_2^4, \overline{x}_4) = \alpha(1,0) = 2$, esto es $\lambda_{24} < \alpha(x_2^4, \overline{x}_4)$, no cumple

Por lo tanto, existe $r = 1$ tal que $\lambda_{21} \geq \alpha(x_2^4, \overline{x}_1)$.

Lema 4.2 *Sea* $\left\{\left(\mathbf{X}^k, \overline{\mathbf{X}}^k\right) \mid k = 1,..., p\right\}$ *el conjunto fundamental de una memoria autoasociativa Alfa-Beta min representada por* $\mathbf{\Lambda}$*, con* $\overline{\mathbf{X}}^k = \tau^e(\mathbf{x}^k, \overline{\mathbf{h}}^k)$ *para* $k = 1, ..., p$*, y sea* $\mathbf{G} = \tau^e(\mathbf{x}^k, \mathbf{w}) \in A^{n+p}$*, una versión alterada con ruido sustractivo de un patrón específico* $\overline{\mathbf{X}}^k$*, siendo* $\mathbf{w} \in A^p$ *un vector cuyas componentes tienen los valores* $w_i = u_i - 1$*, y* $\mathbf{u} \in A^p$ *el vector definido como* $\mathbf{u} = \sum_{i=1}^{p} \mathbf{h}^i$*. Si en la fase de recuperación se presenta* $\mathbf{G}$ *a la entrada de la memoria* $\mathbf{\Lambda}$*, entonces la componente* $\overline{X}_{n+k}^k$ *se recupera de manera correcta; es decir,* $\left(\mathbf{\Lambda} \nabla_\beta \mathbf{G}\right)_{n+k} = \overline{X}_{n+k}^k = 0$.

Demostración.- Esta demostración se hará para dos casos mutuamente exclusivos.

CASO 1 El patrón $\mathbf{G}$ contiene una componente con valor 1. Esto significa que $\exists\, j \in \{1, ..., n+p\}$ tal que $G_j = 1$; además, por la forma de construir el vector $\overline{\mathbf{X}}^k$ es claro que $\overline{X}_{n+k}^k = 0$. Por ello $\alpha(\overline{X}_{n+k}^k, G_j) = \alpha(0,1) = 0$ y, dado que el mínimo valor posible para alguna componente de la memoria $\mathbf{\Lambda}$ es 0, se tiene que $\lambda_{(n+k)j} \geq \alpha(\overline{X}_{n+k}^k, G_j)$. De acuerdo con el Teorema 4.3, $\overline{X}_{n+k}^k$ se recupera de manera correcta.

CASO 2 El patrón $\mathbf{G}$ no contiene una componente con valor 1; es decir $G_j = 0\ \forall\, j \in \{1, ..., n+p\}$. Esto significa que no es posible garantizar la existencia de un valor $j \in \{1, ..., n+p\}$ tal que $\lambda_{(n+k)j} \geq \alpha(\overline{X}_{n+k}^k, G_j)$, y por ello no se puede aplicar el Teorema 4.3. Sin embargo, mostremos la imposibilidad de que $(\mathbf{\Lambda} \nabla_\beta \mathbf{G})_{n+k} = 1$. La fase de recuperación de la memoria autoasociativa Alfa-Beta min $\mathbf{\Lambda}$ con el vector $\mathbf{G}$ a la entrada toma la siguiente forma para la $n+k$-ésima componente recuperada:

$$\left(\mathbf{\Lambda} \nabla_\beta \mathbf{G}\right)_{n+k} = \bigvee_{j=1}^{n} \beta(\lambda_{(n+k)j}, G_j) = \bigvee_{j=1}^{n} \beta\left\{\left[\bigwedge_{\mu=1}^{p} \alpha(\overline{X}_{n+k}^{\mu}, \overline{X}_j^{\mu})\right], G_j\right\}$$

Por la forma de construir el vector $\overline{\mathbf{X}}^k$, además de que $\overline{X}_{n+k}^k = 0$, es importante notar que $\overline{X}_{n+k}^{\mu} \neq 0$, $\forall \mu \neq k$, y de aquí podemos establecer que

$$\bigwedge_{\mu=1}^{p} \alpha(\overline{X}_{n+k}^{\mu}, \overline{X}_j^{\mu}) = \alpha(\overline{X}_{n+k}^k, \overline{X}_j^k) = \alpha(0, \overline{X}_j^k)$$

es diferente de dos sin importar el valor de $\overline{X}_j^k$. Al tomar en cuenta que $G_j = 0\ \forall\, j \in \{1, ..., n+p\}$, podemos concluir que es imposible que

$$\left(\mathbf{\Lambda}\nabla_{\beta}\mathbf{G}\right)_{n+k} = \bigvee_{j=1}^{n} \beta\left(\alpha\left(0, \overline{X}_j^k\right), 0\right)$$

sea uno; es decir, $\left(\mathbf{\Lambda}\nabla_{\beta}\mathbf{G}\right)_{n+k} = 0 = \overline{X}_{n+k}^k$ .■

Ejemplo 4.6 (Ilustra el Caso 1 del Lema 4.2) Tomando el conjunto fundamental del ejemplo 4.1 se construyen los patrones $\overline{\mathbf{X}}^k$ para $k$ = 1, 2, 3, 4 utilizando la transformada vectorial de expansión de los vectores $\mathbf{x}^k$ de la Definición 3:

$$\mathbf{X}^1 = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 0 \\ 1 \\ 1 \\ 1 \end{bmatrix}, \quad \mathbf{X}^2 = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 1 \\ 1 \end{bmatrix}, \quad \mathbf{X}^3 = \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \\ 1 \\ 1 \\ 0 \\ 1 \end{bmatrix}, \quad \mathbf{X}^4 = \begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 1 \\ 1 \\ 1 \\ 0 \end{bmatrix}$$

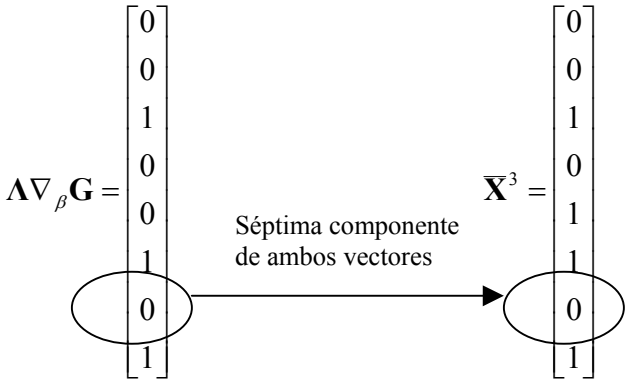
La matriz de la memoria autoasociativa Alfa-Beta *min* es:

$$\mathbf{\Lambda} = \begin{bmatrix} 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 \end{bmatrix}$$

Ahora, tomando $k$ = 3 utilizaremos vector $\overline{\mathbf{X}}^3$ y obtendremos su versión ruidosa **G**, con ruido sustractivo:

$$\overline{\mathbf{X}}^3 = \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \\ 1 \\ 1 \\ 0 \\ 1 \end{bmatrix}, \text{ el vector ruidoso, con ruido sustractivo, } \mathbf{G} \text{ es: } \mathbf{G} = \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

Al presentarle **G** a la matriz $\mathbf{\Lambda}$, el vector recuperado es:

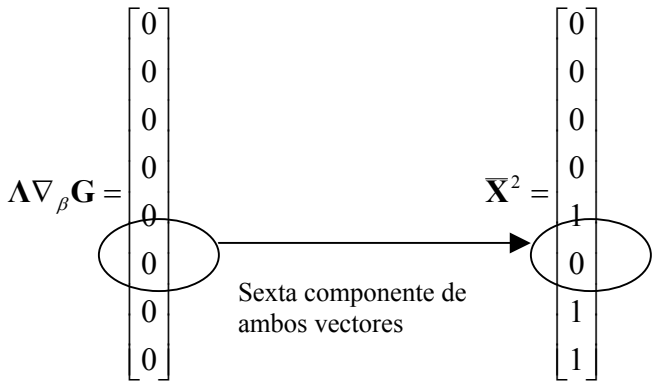



Recordemos que en este ejemplo $n = 4$ y $k = 3$, entonces $\left(\mathbf{\Lambda}\nabla_\beta\mathbf{G}\right)_{4+3} = \overline{X}^3_{4+3} = 0$. Por lo tanto, se recupera de manera correcta la séptima componente del tercer patrón.

Ejemplo 4.7 (Ilustra el Caso 2 del Lema 4.2) Utilizando la matriz $\mathbf{\Lambda}$ obtenida en el ejemplo 4.6 y con $k = 2$ se obtiene el vector **G** que es la versión ruidosa, con ruido sustractivo, de $\overline{\mathbf{X}}^2$, cuyos valores de componentes son:

$$\overline{\mathbf{X}}^2 = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 1 \\ 1 \end{bmatrix}, \text{ el vector ruidoso, con ruido sustractivo, } \mathbf{G} \text{ es: } \mathbf{G} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

Al presentarle **G** a la matriz $\mathbf{\Lambda}$, el vector recuperado es:




Para este ejemplo $n = 4$ y $k = 2$, entonces $\left(\mathbf{\Lambda}\nabla_\beta\mathbf{G}\right)_{4+2} = X^2_{4+2} = 0$. Por lo tanto, se recupera de manera correcta la sexta componente del segundo patrón.

Teorema 4.4 *Sea* $\left\{\left(\mathbf{X}^k, \mathbf{X}^k\right) \mid k = 1,..., p\right\}$ *el conjunto fundamental de una memoria autoasociativa Alfa-Beta min representada por* $\mathbf{\Lambda}$*, con* $\overline{\mathbf{X}}^k = \tau^e(\mathbf{x}^k, \overline{\mathbf{h}}^k)$ *para* $k = 1, ..., p$*, y sea* $\mathbf{G} = \tau^e(\mathbf{x}^k, \mathbf{w}) \in A^{n+p}$*, una versión alterada con ruido sustractivo de un patrón específico* $\overline{\mathbf{X}}^k$*, siendo* $\mathbf{w} \in A^p$ *un vector cuyas componentes tienen los valores* $w_i = u_i - 1$*, y* $\mathbf{u} \in A^p$ *el vector definido como* $\mathbf{u} = \sum_{i=1}^{p} \mathbf{h}^i$*. Asumamos que en la fase de recuperación se presenta* $\mathbf{G}$ *a la entrada de la memoria* $\mathbf{\Lambda}$ *y se obtiene el patrón* $\mathrm{S} = \mathbf{\Lambda}\nabla_\beta \mathbf{G} \in A^{n+p}$*. Si al tomar como argumento el vector* $\mathbf{S}$ *se realiza la transformada vectorial de contracción* $\mathbf{s} = \tau^c(\mathbf{S}, n) \in A^p$*, entonces el vector* $\mathbf{s}$ *tiene dos posibilidades mutuamente exclusivas: o sucede que* $\exists\, k \in \{1, ..., p\}$ *tal que* $\mathbf{s} = \overline{\mathbf{h}}^k$*, o* $\mathbf{s}$ *no es un vector zero-hot.*

Demostración.- Por la definición de transformada vectorial de contracción se tiene que $s_i = S_{i+n} = (\mathbf{\Lambda}\nabla_\beta \mathbf{G})_{i+n}$ para $1 \le i \le p$, y en particular haciendo $i = k$ se tiene que $s_k = S_{k+n} = (\mathbf{\Lambda}\nabla_\beta \mathbf{G})_{k+n}$. Pero por el Lema 4.2 resulta que $\left(\mathbf{\Lambda}\nabla_\beta\mathbf{G}\right)_{n+k} = \overline{X}^k_{n+k}$, y dado que $\overline{\mathbf{X}}^k = \tau^e(\mathbf{x}^k, \overline{\mathbf{h}}^k)$, el valor $\overline{X}^k_{n+k}$ es igual al valor de la componente $\overline{h}^k_k = 0$ ; es decir, $s_k = 0$. Al considerar que $s_k = 0$, el vector $\mathbf{s}$ tiene dos posibilidades mutuamente exclusivas: o sucede que $s_j = 1\ \forall\ j \ne k$ en cuyo caso $\mathbf{s} = \overline{\mathbf{h}}^k$; o bien, ocurre que $\exists\ j \in \{1, ..., p\}$, $j \ne k$ para el cual $s_j = 0$, en cuyo caso no es posible que $\mathbf{s}$ sea un vector *zero-hot*, atendiendo a la definición 2. ■

Ejemplo 4.8 (Ilustra el Teorema 4.4) Tomando la matriz $\mathbf{\Lambda}$ obtenida en el ejemplo 4.6 y con $k = 1$, los valores de las componentes $\overline{\mathbf{X}}^1$ y su respectivo patrón ruidoso $\mathbf{G}$, con ruido sustractivo, son

$$\mathbf{X}^1 = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 0 \\ 1 \\ 1 \\ 1 \end{bmatrix} \quad \text{y} \quad \mathbf{G} = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

Al presentarle **G** a la matriz $\mathbf{\Lambda}$ se obtiene el vector **S**:

$$\mathbf{S} = \mathbf{\Lambda}\nabla_{\beta}\mathbf{G} = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 0 \\ 1 \\ 1 \\ 1 \end{bmatrix}$$

Al tomar como argumento este vector y realizar la transformada vectorial de contracción del vector **S**, obtenemos el vector **s**,

$$\mathbf{s} = \begin{bmatrix} 0 \\ 1 \\ 1 \\ 1 \end{bmatrix}$$

y de acuerdo a la Definición 2, podemos observar que **s** es el primer vector *zero-hot* de 4 bits.

Ahora realicemos el mismo proceso pero para $k = 4$. Entonces, $\overline{\mathbf{X}}^4$ y su respectivo patrón ruidoso **G**, con ruido sustractivo, son:

$$\mathbf{X}^4 = \begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 1 \\ 1 \\ 1 \\ 0 \end{bmatrix} \quad \text{y} \quad \mathbf{G} = \begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

Al presentarle **G** a la matriz $\mathbf{\Lambda}$ se obtiene el vector **S**:

$$\mathbf{S} = \mathbf{\Lambda}\nabla_{\beta}\mathbf{G} = \begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \\ 0 \end{bmatrix}$$

Al tomar como argumento este vector y realizar la transformada vectorial de contracción del vector **S**, obtenemos el vector **s**,

$$\mathbf{r} = \begin{bmatrix} 0 \\ 1 \\ 1 \\ 0 \end{bmatrix}$$

De acuerdo a la Definición 2 **s** no es un vector *zero-hot*.

Lema 4.3 *Sea* $\{(\mathbf{X}^k, \mathbf{X}^k) \mid k = 1, ..., p\}$ *el conjunto fundamental de una memoria autoasociativa Alfa-Beta max representada por* **V**, *con* $\mathbf{X}^k = \tau^{\varepsilon}(\mathbf{x}^k, \mathbf{h}^k) \in A^{n+p}\ \forall\ k \in \{1, ..., p\}$. *Si t es un índice tal que* $n+1 \leq t \leq n+p$ *entonces* $v_{ij} \neq 0\ \forall\ j \in \{1, ...,n+ p\}$.

Demostración.- Para establecer que $v_{ij} \neq 0\ \forall\ j \in \{1, ...,n+ p\}$, en virtud de la definición de $\alpha$ basta con encontrar, para cada $\forall\ t \in \{n+1, ...,n+ p\}$, un índice $\mu$ para el cual $X_t^{\mu} = 1$ en la expresión que permite obtener la componente *tj*-ésima de la memoria **V**, la cual es $v_{tj} = \vee_{\mu=1}^{p} \alpha(X_t^{\mu}, X_j^{\mu})$. En efecto, por la forma de construir cada vector $\mathbf{X}^{\mu} = \tau^{\varepsilon}(\mathbf{x}^{\mu}, \mathbf{h}^{\mu})$ para $\mu = 1, ..., p$, y dado el dominio del índice $t \in \{n+1, ...,n+ p\}$, para cada *t* existe $s \in \{1, ..., p\}$ tal que $t = n + s$; por ello dos valores útiles para establecer el resultado son $\mu = s$ y $t = n$

+ $s$, porque $X^s_{n+s} = 1$, entonces $\nu_{tj} = \vee^p_{\mu=1} \alpha(X^\mu_t, X^\mu_j) = \alpha(X^s_{n+s}, X^s_j) = \alpha(1, X^s_j)$ , valor que es diferente de cero; es decir, $\nu_{ij} \neq 0 \ \forall \ j \in \{1, ...,n+ p\}$. ■

Ejemplo 4.9 (Ilustra el Lema 4.3) Tomemos los vectores $\mathbf{X}^k$ para $k$ = 1, 2, 3, 4 de ejemplo 4.2

$$\mathbf{X}^1 = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}, \quad \mathbf{X}^2 = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \end{bmatrix}, \quad \mathbf{X}^3 = \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \end{bmatrix}, \quad \mathbf{X}^4 = \begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}$$

La matriz de la memoria autoasociativa Alfa-Beta *max* es:

$$\mathbf{V} = \left.\begin{bmatrix} 1 & 1 & 2 & 2 & 2 & 2 & 2 & 1 \\ 1 & 1 & 2 & 2 & 2 & 2 & 2 & 2 \\ 2 & 2 & 1 & 2 & 2 & 2 & 2 & 2 \\ 1 & 1 & 1 & 1 & 1 & 2 & 2 & 2 \\ 1 & 1 & 1 & 1 & 1 & 2 & 2 & 2 \\ 2 & 2 & 2 & 2 & 2 & 1 & 2 & 2 \\ 2 & 2 & 1 & 2 & 2 & 2 & 1 & 2 \\ 1 & 1 & 2 & 2 & 2 & 2 & 2 & 1 \end{bmatrix}\right\} \text{ Para } t \in \{5,6,7,8\} \text{ y } \forall \ j \in \{1, 2, .., 8\} \ \nu_{ij} \neq 0$$

Lema 4.4 *Sea* $\{(\mathbf{X}^k, \mathbf{X}^k) \mid k = 1, ..., p\}$ *el conjunto fundamental de una memoria autoasociativa Alfa-Beta max representada por* $\mathbf{V}$*, con* $\mathbf{X}^k = \tau^e(\mathbf{x}^k, \mathbf{h}^k)$ *para* $k = 1, ..., p$*, y sea* $\mathbf{F} = \tau^e(\mathbf{x}^k, \mathbf{u}) \in A^{n+p}$ *una versión alterada con ruido aditivo de un patrón específico* $\mathbf{X}^k$*, siendo* $\mathbf{u} \in A^p$ *el vector definido como* $\mathbf{u} = \sum^p_{i=1} \mathbf{h}^i$*. Asumamos que en la fase de recuperación se presenta* $\mathbf{F}$ *a la entrada de la memoria* $\mathbf{V}$*. Dado un índice* $t \in \{n+1, ...,n+ p\}$ *fijo tal que* $t \neq n + k$*, se cumple que* $(\mathbf{V} \ \Delta_\beta \ \mathbf{F})_t = 1$ *si y sólo si la siguiente proposición lógica es verdadera* $\forall \ j \in \{1, ...,n+ p\}$ $(F_j = 0 \rightarrow \nu_{tj} = 2)$.

Demostración.- Por la forma de construir los vectores $\mathbf{X}^k = \tau^e(\mathbf{x}^k, \mathbf{h}^k)$ y $\mathbf{F} = \tau^e(\mathbf{x}^k, \mathbf{u})$, se tiene que $F_t = 1$ es la componente con ruido aditivo de la componente $X^k_t = 0$.

$\Rightarrow$) Hay dos casos posibles:

CASO 1 El patrón **F** no contiene componentes con valor 0; es decir $F_j = 1$ $j \in \{1, ...,n+ p\}$. Esto significa que el antecedente de la proposición lógica $F_j = 0 \rightarrow \nu_{tj} = 2$ es falso, y por ello, sin importar el valor de verdad del consecuente $\nu_{tj} = 2$, la expresión $\forall\, j \in \{1, ...,n+ p\}$ $(F_j = 0 \rightarrow \nu_{tj} = 2)$ es verdadera.

CASO 2 El patrón **F** contiene al menos una componente con valor 0; es decir, $\exists\, r \in \{1, ..., p\}$ tal que $F_r = 0$. Por hipótesis $(\mathbf{V}\ \Delta_\beta\ \mathbf{F})_t = 1$, lo cual significa que la condición de recuperación correcta de $X_t^k = 0$ no se cumple; es decir, de acuerdo con el Teorema 4.1 la expresión $\neg[\exists\ j \in \{1, ...,n+ p\}$ tal que $\nu_{tj} \le \alpha(X_t^k, F_j)]$ es verdadera, lo cual es equivalente a la afirmación

$$\forall\, j \in \{1, ...,n+ p\} \text{ se cumple que } \nu_{tj} \le \alpha(X_t^k, F_j)$$

En particular, para $j = r$, y considerando que $X_t^k = 0$, esta desigualdad queda así: $\nu_{tr} > \alpha(X_t^k, F_r) = \alpha(0,0) = 1$. Es decir, $\nu_{tr} = 2$, y por ello la expresión $\forall\, j \in \{1, ...,n+ p\}$ $(F_j = 0 \rightarrow \nu_{tj} = 2)$ es verdadera.

$\Leftarrow$) Supongamos que la siguiente expresión es verdadera $\forall\, j \in \{1, ...,n+ p\}$ $(F_j = 0 \rightarrow \nu_{tj} = 2)$. Hay dos casos posibles:

CASO 1 El patrón **F** no contiene componentes con valor 0; es decir $F_j = 1$ $\forall\, j \in \{1, ...,n+ p\}$. Al considerar que $\left(\mathbf{V}\Delta_\beta\mathbf{F}\right)_t = \wedge_{j=1}^{n+p}\beta\left(\nu_{tj}, F_j\right)$, de acuerdo con la definición de $\beta$ basta con mostrar que $\forall\, j \in \{1, ...,n+ p\}$ $\nu_{tj} \neq 0$, lo cual queda garantizado por el Lema 4.3. Así, queda demostrado que $\left(\mathbf{V}\Delta_\beta\mathbf{F}\right)_t = \wedge_{j=1}^{n+p}\beta\left(\nu_{tj}, F_j\right) = \wedge_{j=1}^{n+p}\beta\left(\nu_{tj}, 1\right) = 1$.

CASO 2 El patrón **F** contiene al menos una componente con valor 0; es decir, $\exists\, r \in \{1, ..., n+p\}$ tal que $F_r = 0$. Por hipótesis, se tiene que $\forall\, j \in \{1, ...,n+ p\}$ $(F_j = 0 \rightarrow \nu_{tj} = 2)$ y, en particular, para $j = r$ y $\nu_{tr} = 2$, lo cual significa que $\left(\mathbf{V}\Delta_\beta\mathbf{F}\right)_t = \wedge_{j=1}^{n+p}\beta\left(\nu_{tj}, F_j\right) = \beta\left(\nu_{tr}, F_r\right) = \beta(2,1) = 1$.■

Ejemplo 4.10 (Ilustra el Caso 2 del condicional $\Rightarrow$ del Lema 4.4) Utilicemos el vector $\mathbf{X}^3$, con $k = 3$, del ejemplo 4.9 y construyamos su vector ruidoso **F**, con ruido aditivo

$$\mathbf{X}^3 = \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \end{bmatrix}, \quad \mathbf{F} = \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}$$

Al presentarle **F** a **V**, presentada en el mismo ejemplo, se obtiene:

$$\mathbf{V}\Delta_\beta\mathbf{F} = \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 1 \\ 1 \\ 0 \end{bmatrix} \rightarrow (\mathbf{V}\,\Delta_\beta\,\mathbf{F})_6 = 1 \quad t = 6$$

$$\mathbf{F} = \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} \qquad \mathbf{V} = \begin{bmatrix} 1\ 1\ 2\ 2\ 2\ 2\ 2\ 1 \\ 1\ 1\ 2\ 2\ 2\ 2\ 2\ 2 \\ 2\ 2\ 1\ 2\ 2\ 2\ 2\ 2 \\ 1\ 1\ 1\ 1\ 1\ 2\ 2\ 2 \\ 1\ 1\ 1\ 1\ 1\ 2\ 2\ 2 \\ 2\ 2\ 2\ 2\ 2\ 1\ 2\ 2 \\ 2\ 2\ 1\ 2\ 2\ 2\ 1\ 2 \\ 1\ 1\ 2\ 2\ 2\ 2\ 2\ 1 \end{bmatrix} \quad t = 6$$

$$2\ \ 2\ \ 2\ \ 2\ \ 2\ \ 1\ \ 2\ \ 2$$

Corolario 4.1 *Sea* $\{(\mathbf{X}^k, \mathbf{X}^k) \mid k = 1, ..., p\}$ *el conjunto fundamental de una memoria autoasociativa Alfa-Beta max representada por* **V**, *con* $\mathbf{X}^k = \tau^e(\mathbf{x}^k, \mathbf{h}^k)$ *para* $k = 1, ..., p$, *y sea* $\mathbf{F} = \tau^e(\mathbf{x}^k, \mathbf{u}) \in A^{n+p}$ *una versión alterada con ruido aditivo de un patrón específico* $\mathbf{X}^k$, *siendo* $\mathbf{u} \in A^p$ *el vector definido como* $\mathbf{u} = \sum_{i=1}^{p} \mathbf{h}^i$ . *Asumamos que en la fase de recuperación se presenta* **F** *a la entrada de la memoria* **V**. *Dado un índice* $t \in \{n+1, ..., n+p\}$ *fijo tal que* $t \neq n + k$, *se cumple que* $(\mathbf{V}\ \Delta_\beta\ \mathbf{F})_t = 0$ *si y sólo si la siguiente proposición lógica es verdadera* $\forall\ j \in \{1, ..., n+p\}$ $(F_j = 0$ AND $v_{tj} \neq 2)$.

Demostración.- En general, dadas dos proposiciones lógicas $P$ y $Q$, la proposición lógica ($P$ si y sólo si $Q$) es equivalente a la proposición ($\neg P$ si y sólo si $\neg Q$). Si se identifica $P$ con la igualdad $(\mathbf{V}\ \Delta_\beta\ \mathbf{F})_t = 1$ y $Q$ con la expresión $\forall\ j \in \{1, ...,n+ p\}\ \ (F_j = 0 \rightarrow \nu_{tj} = 2)$, por Lema 4.4 la siguiente proposición es verdadera $\{\neg\ [(\mathbf{V}\ \Delta_\beta\ \mathbf{F})_t = 1]$ si y sólo si $\neg[\forall\ j \in \{1, ...,n+ p\}\ \ (F_j = 0 \rightarrow \nu_{tj} = 2)]\}$ . Esta expresión se transforma en las siguientes proposiciones equivalentes

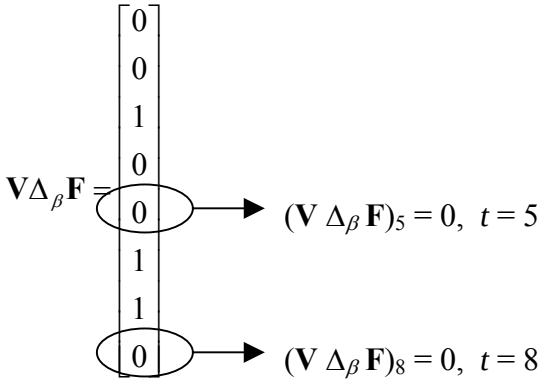
$$\{(\mathbf{V}\ \Delta_\beta\ \mathbf{F})_t = 0 \text{ si y solo si } \exists\ j \in \{1, ...,n+ p\} \text{ tal que } \neg\ (F_j = 0 \rightarrow \nu_{tj} = 2)\}$$

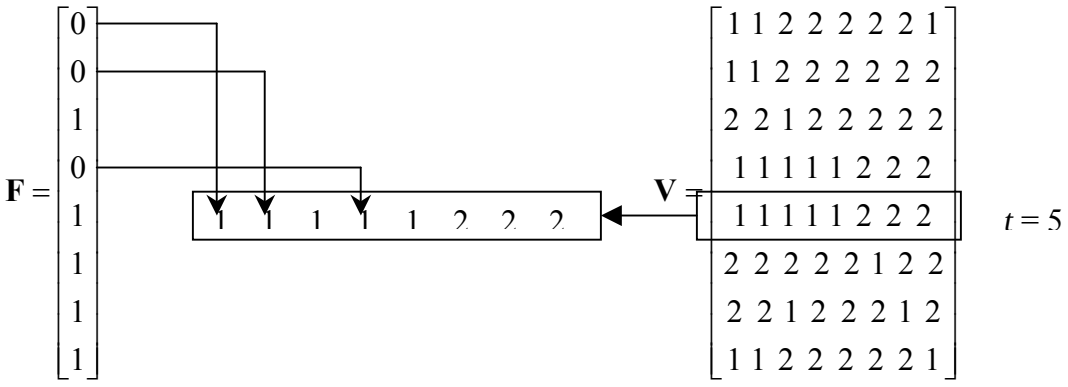
$$\{(\mathbf{V}\ \Delta_\beta\ \mathbf{F})_t = 0 \text{ si y solo si } \exists\ j \in \{1, ...,n+ p\} \text{ tal que } \neg[\neg(F_j = 0) \text{ OR } \nu_{tj} = 2]\}$$

$$\{(\mathbf{V}\ \Delta_\beta\ \mathbf{F})_t = 0 \text{ si y solo si } \exists\ j \in \{1, ...,n+ p\} \text{ tal que } [\neg[\neg(F_j = 0)] \text{ AND } \neg\ (\nu_{tj} = 2)]\}$$

$$\{(\mathbf{V}\ \Delta_\beta\ \mathbf{F})_t = 0 \text{ si y solo si } \exists\ j \in \{1, ...,n+ p\} \text{ tal que } [(F_j = 0) \text{ AND } \nu_{tj} \neq 2]\}\ \blacksquare$$

Ejemplo 4.11 (Ilustra el Corolario 4.1) Tomando $\mathbf{X}^3$ y $\mathbf{F}$ del ejemplo 4.10, al presentarle $\mathbf{F}$ a $\mathbf{V}$, tenemos:



Lema 4.5 *Sea Sea* $\left\{\left(\overline{\mathbf{X}}^k, \overline{\mathbf{X}}^k\right) \mid k = 1,..., p\right\}$ *el conjunto fundamental de una memoria autoasociativa Alfa-Beta min representada por* $\mathbf{\Lambda}$*, con* $\overline{\mathbf{X}}^k = \tau^e(\mathbf{x}^k, \overline{\mathbf{h}}^k) \in A^{n+p}$ $\forall$ k $\in$ {1, ..., *p*}. *Si t es un índice tal que n*+1 ≤ *t* ≤ *n*+*p entonces* $\lambda_{tj} \neq 2$ $\forall$ *j* $\in$ {1, ..., *n*+*p*}.

Demostración.- Para establecer que $\lambda_{tj} \neq 2$ $\forall$ *j* $\in$ {1, ..., *n*+*p*}, en virtud de la definición de $\alpha$, basta con encontrar, para cada *t* $\in$ {*n*+1, ..., *n*+*p*}, un índice $\mu$ para el cual $\overline{X}_t^{\mu} = 0$ en la expresión que permite obtener la componente *tj*-ésima de la memoria $\mathbf{\Lambda}$, la cual es $\lambda_{tj} = \wedge_{\mu=1}^{p} \alpha(\overline{X}_t^{\mu}, \overline{X}_j^{\mu})$. En efecto, por la forma de construir cada vector $\overline{\mathbf{X}}^k = \tau^e(\mathbf{x}^k, \overline{\mathbf{h}}^k)$ para $\mu$ = 1, ..., *p*, y dado el dominio del índice *t* $\in$ {*n*+1, ..., *n*+*p*}, para cada *t* existe *s* $\in$ {1, ..., *p*} tal que *t* = *n* + *s*; por ello dos valores útiles para establecer el resultado son $\mu$ = *s* y *t* = *n* + *s*, porque, $\overline{X}_{n+s}^{s} = 0$, entonces $\lambda_{tj} = \wedge_{\mu=1}^{p} \alpha(\overline{X}_t^{\mu}, \overline{X}_j^{\mu}) = \alpha(\overline{X}_{n+s}^{s}, \overline{X}_j^{s}) = \alpha(0, X_j^{\mu})$, valor que es diferente de dos; es decir, $\lambda_{tj} \neq 2$ $\forall$ *j* $\in$ {1, ..., *n*+*p*}.■

Ejemplo 4.12 (Ilustra el Lema 4.5) Tomemos los vectores $\overline{\mathbf{X}}^k$ para *k* = 1, 2, 3, 4 de ejemplo 4.6

$$\overline{\mathbf{X}}^1 = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 0 \\ 1 \\ 1 \\ 1 \end{bmatrix}, \quad \overline{\mathbf{X}}^2 = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 1 \\ 1 \end{bmatrix}, \quad \overline{\mathbf{X}}^3 = \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \\ 1 \\ 1 \\ 0 \\ 1 \end{bmatrix}, \quad \overline{\mathbf{X}}^4 = \begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 1 \\ 1 \\ 1 \\ 0 \end{bmatrix}$$

La matriz de la memoria autoasociativa Alfa-Beta *min* es:

$$\mathbf{\Lambda} = \left.\begin{bmatrix} 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 \end{bmatrix}\right\} \text{ Para } t \in \{5,6,7,8\} \text{ y } \forall\, j \in \{1, 2, .., 8\}\ \lambda_{ij} \neq 2$$

Lema 4.6 *Sea* $\left\{\left(\overline{\mathbf{X}}^k, \mathbf{X}^k\right) \mid k = 1,..., p\right\}$ *el conjunto fundamental de una memoria autoasociativa Alfa-Beta min representada por* $\mathbf{\Lambda}$*, con* $\overline{\mathbf{X}}^k = \tau^e(\mathbf{x}^k, \overline{\mathbf{h}}^k)$ *para* $k = 1, ..., p$*, y sea* $\mathbf{G} = \tau^e(\mathbf{x}^k, \mathbf{w}) \in A^{n+p}$*, una versión alterada con ruido sustractivo de un patrón específico* $\mathbf{X}^k$*, siendo* $\mathbf{w} \in A^p$ *un vector cuyas componentes tienen los valores* $w_i = u_i - 1$*, y* $\mathbf{u} \in A^p$ *el vector definido como* $\mathbf{u} = \sum_{i=1}^{p} \mathbf{h}^i$*. Asumamos que en la fase de recuperación se presenta* $\mathbf{G}$ *a la entrada de la memoria* $\mathbf{\Lambda}$*. Dado un índice* $t \in \{n+1, ..., n+p\}$ *fijo tal que* $t \neq n + k$ *se cumple que* $(\mathbf{\Lambda} \nabla_\beta \mathbf{G})_t = 0$*, si y sólo si la siguiente proposición lógica es verdadera* $\forall\, j \in \{1, ..., n+p\}$ $(G_j = 1 \rightarrow \lambda_{tj} = 0)$.

Demostración.- Por la forma de construir los vectores $\overline{\mathbf{X}}^k = \tau^e(\mathbf{x}^k, \overline{\mathbf{h}}^k)$ y $\mathbf{G} = \tau^e(\mathbf{x}^k, \mathbf{w})$, se tiene que $G_t = 1$ es la componente con ruido sustractivo de la componente $\overline{X}_t^k = 1$.

$\Rightarrow$) Hay dos casos posibles:

CASO 1 El patrón $\mathbf{G}$ no contiene componentes con valor 1; es decir $G_j = 0 \;\; \forall\, j \in \{1, ..., n+p\}$. Esto significa que el antecedente de la proposición lógica $G_j = 1 \rightarrow \lambda_{tj} = 0$ es falso, y por ello, sin importar el valor de verdad del consecuente $\lambda_{tj} = 0$, la expresión $\forall\, j \in \{1, ..., n+p\}$ $(G_j = 1 \rightarrow \lambda_{tj} = 0)$ es verdadera.

CASO 2 El patrón $\mathbf{G}$ contiene al menos una componente con valor 1; es decir, $\exists\, r \in \{1, ..., n+p\}$ tal que $G_r = 1$. Por hipótesis $(\mathbf{\Lambda} \nabla_\beta \mathbf{G})_t = 0$, lo cual significa que la condición de recuperación correcta de $\overline{X}_t^k = 1$ no se cumple; es decir, de acuerdo con el Teorema 4.3 la expresión $\neg[\exists\, j \in \{1, ..., n+p\}$ tal que $\lambda_{tj} \geq \alpha(\overline{X}_t^k, G_j)\,]$ es verdadera, lo cual es equivalente a la afirmación

$$\forall\, j \in \{1, ..., n+p\} \;\text{ se cumple que }\; \lambda_{tj} < \alpha(\overline{X}_t^k, G_j)$$

En particular, para $j = r$, y considerando que $\overline{X}_t^k = 1$, esta desigualdad queda así: $\lambda_{tr} < \alpha(\overline{X}_t^k, G_r) = \alpha(1,1) = 1$ . Es decir, $\lambda_{tr} = 0$, y por ello la expresión $\forall\, j \in \{1, ..., n+p\}$ $(G_j = 1 \rightarrow \lambda_{tj} = 0)$ es verdadera.

$\Leftarrow$) Supongamos que la siguiente expresión es verdadera $\forall\, j \in \{1, ..., n+p\}$ $(G_j = 1 \rightarrow \lambda_{tj} = 0)$. Hay dos casos posibles:

CASO 1 El patrón $\mathbf{G}$ no contiene componentes con valor 1; es decir $G_j = 0$ $\forall\, j \in \{1, ..., n+p\}$. Al considerar que $\left(\mathbf{\Lambda} \nabla_\beta \mathbf{G}\right)_t = \vee_{j=1}^{n+p} \beta\left(\lambda_{tj}, G_j\right)$, de acuerdo con la definición de $\beta$ basta con mostrar que $\forall\, j \in \{1, ..., n+p\}$ $\lambda_{tj} \neq$, lo cual queda garantizado por el Lema 4.5. Así, queda demostrado que

$$\left(\mathbf{\Lambda} \nabla_\beta \mathbf{G}\right)_t = \vee_{j=1}^{n+p} \beta\left(\lambda_{tj}, G_j\right) = \vee_{j=1}^{n+p} \beta\left(\lambda_{tj}, 0\right) = 0\,.$$

CASO 2 El patrón **G** contiene al menos una componente con valor 1; es decir, $\exists\, r \in \{1, ..., n+p\}$ tal que $G_r = 1$. Por hipótesis, se tiene que $\forall\, j \in \{1, ..., n+p\}$ $(G_j = 1 \rightarrow \lambda_{tj} = 0)$ y, en particular, para $j = r$ y $\lambda_{tr} = 0$, lo cual significa que $\left(\mathbf{\Lambda} \nabla_\beta \mathbf{G}\right)_t = \vee_{j=1}^{n+p} \beta\left(\lambda_{tj}, G_j\right) = \beta\left(\lambda_{tr}, G_r\right) = \beta(0,0) = 0$ . ■

Ejemplo 4.13 (Ilustra el Caso 2 del condicional $\Rightarrow$ del Lema 4.6) Utilicemos el vector $\overline{\mathbf{X}}^4$, con $k = 4$, del ejemplo 4.12 y construyamos su vector ruidoso **G**, con ruido sustractivo

$$\overline{\mathbf{X}}^4 = \begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 1 \\ 1 \\ 1 \\ 0 \end{bmatrix}, \quad \mathbf{G} = \begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

Al presentarle **G** a $\mathbf{\Lambda}$, presentada en el mismo ejemplo, se obtiene:

$$\mathbf{\Lambda} \nabla_\beta \mathbf{G} = \begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \\ 0 \end{bmatrix} \rightarrow (\mathbf{\Lambda} \nabla_\beta \mathbf{G})_5 = 0 \quad t = 5$$

$$\mathbf{G} = \begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \qquad \mathbf{\Lambda} = \begin{bmatrix} 1\ 1\ 0\ 1\ 0\ 0\ 0\ 0 \\ 1\ 1\ 0\ 1\ 0\ 0\ 0\ 0 \\ 0\ 0\ 1\ 1\ 0\ 0\ 0\ 0 \\ 0\ 0\ 0\ 1\ 0\ 0\ 0\ 0 \\ 0\ 0\ 0\ 0\ 1\ 0\ 0\ 0 \\ 1\ 1\ 1\ 1\ 0\ 1\ 0\ 0 \\ 1\ 1\ 0\ 1\ 0\ 0\ 1\ 0 \\ 0\ 0\ 1\ 1\ 0\ 0\ 0\ 1 \end{bmatrix} \qquad t = 5$$

$$0\ \ 0\ \ 0\ \ 0\ \ 1\ \ 0\ \ 0\ \ 0$$

Corolario 4.2 *Sea* $\left\{\left(\overline{\mathbf{X}}^k, \overline{\mathbf{X}}^k\right) \mid k = 1,..., p\right\}$ *el conjunto fundamental de una memoria autoasociativa Alfa-Beta min representada por* $\mathbf{\Lambda}$*, con* $\overline{\mathbf{X}}^k = \tau^e(\mathbf{x}^k, \overline{\mathbf{h}}^k)$ *para* $k = 1, ..., p$*, y sea* $\mathbf{G} = \tau^e(\mathbf{x}^k, \mathbf{w}) \in A^{n+p}$*, una versión alterada con ruido sustractivo de un patrón específico* $\mathbf{X}^k$*, siendo* $\mathbf{w} \in A^p$ *un vector cuyas componentes tienen los valores* $w_i = u_i - 1$*, y* $\mathbf{u} \in A^p$ *el vector definido como* $\mathbf{u} = \sum_{i=1}^{p} \mathbf{h}^i$*. Asumamos que en la fase de recuperación se presenta* $\mathbf{G}$ *a la entrada de la memoria* $\mathbf{\Lambda}$*. Dado un índice* $t \in \{n+1, ..., n+p\}$ *fijo tal que* $t \neq n + k$ *se cumple que* $(\mathbf{\Lambda} \nabla_\beta \mathbf{G})_t = 1$*, si y sólo si la siguiente proposición lógica es verdadera* $\exists\, j \in \{1, ..., n+p\}$ $(G_j = 1$ AND $\lambda_{tj} \neq 0)$.

Demostración.- En general, dadas dos proposiciones lógicas $P$ y $Q$, la proposición lógica ( $P$ si y sólo si $Q$) es equivalente a la proposición ($\neg P$ si y sólo si $\neg Q$). Si se identifica $P$ con la igualdad $(\mathbf{\Lambda} \nabla_\beta \mathbf{G})_t = 0$ y $Q$ con la expresión $\forall\, j \in \{1, ..., n+p\}$ $(G_j = 1 \rightarrow \lambda_{tj} = 0)$, por Lema 4.6 la siguiente proposición es verdadera $\{\neg[(\mathbf{\Lambda} \nabla_\beta \mathbf{G})_t = 0]$ si y sólo si $\neg[\forall\, j \in \{1, ..., n+p\}$ $(G_j = 1 \rightarrow \lambda_{tj} = 0)]\}$. Esta expresión se transforma en las siguientes proposiciones equivalentes:

$$\{(\mathbf{\Lambda} \nabla_\beta \mathbf{G})_t = 1 \text{ si y sólo si } \exists\, j \in \{1, ..., n+p\} \text{ tal que } \neg\, (G_j = 1 \rightarrow \lambda_{tj} = 0)]\}$$

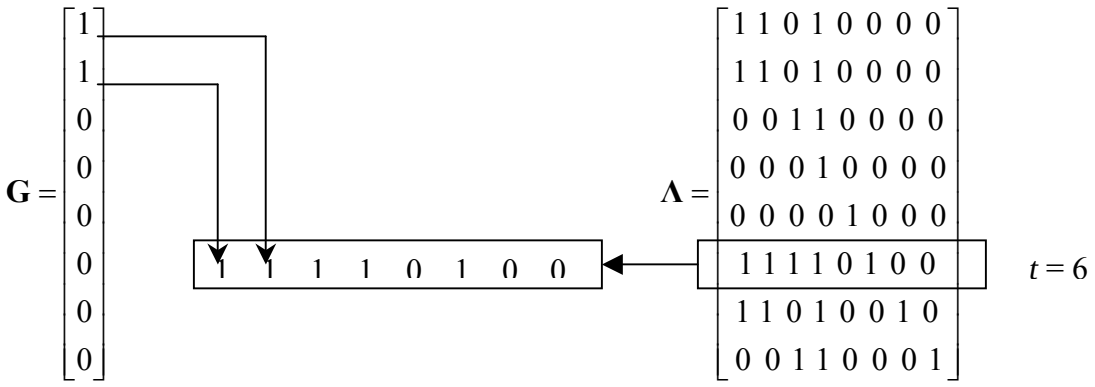
$$\{(\mathbf{\Lambda} \nabla_\beta \mathbf{G})_t = 1 \text{ si y sólo si } \exists\, j \in \{1, ..., n+p\} \text{ tal que } \neg[\neg(G_j = 1) \text{ OR } \lambda_{tj} = 0]\}$$

$$\{(\mathbf{\Lambda} \nabla_\beta \mathbf{G})_t = 1 \text{ si y sólo si } \exists\, j \in \{1, ..., n+p\} \text{ tal que } [\neg[\neg(G_j = 1)] \text{ AND } \neg(\lambda_{tj} = 0)]\}$$

$$\{(\mathbf{\Lambda} \nabla_\beta \mathbf{G})_t = 1 \text{ si y sólo si } \exists\, j \in \{1, ..., n+p\} \text{ tal que } [G_j = 1 \text{ AND } \lambda_{tj} \neq 0]\}.\ \blacksquare$$

Ejemplo 4.14 (Ilustra el Corolario 4.2) Tomando el $\mathbf{X}^4$ y $\mathbf{G}$ del ejemplo 4.13, al presentarle $\mathbf{G}$ a $\mathbf{\Lambda}$,

$$\mathbf{\Lambda} \nabla_\beta \mathbf{G} = \begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \\ 0 \end{bmatrix} \begin{array}{l} \\ \\ \\ \\ \\ \rightarrow (\mathbf{\Lambda} \nabla_\beta \mathbf{G})_6 = 1 \quad t = 6 \\ \rightarrow (\mathbf{\Lambda} \nabla_\beta \mathbf{G})_7 = 1 \quad t = 7 \\ \\ \end{array}$$




Lema 4.7 *Sea* $\{(\mathbf{X}^k, \mathbf{X}^k) \mid k = 1, ..., p\}$ *el conjunto fundamental de una memoria autoasociativa Alfa-Beta max representada por* $\mathbf{V}$*, con* $\mathbf{X}^k = \tau^e(\mathbf{x}^k, \mathbf{h}^k)\ \forall\ k \in \{1, ..., p\}$ *y sea* $\left\{\left(\overline{\mathbf{X}}^k, \overline{\mathbf{X}}^k\right) \mid k = 1,..., p\right\}$ *el conjunto fundamental de una memoria autoasociativa Alfa-Beta min representada por* $\mathbf{\Lambda}$*, con* $\overline{\mathbf{X}}^k = \tau^e(\mathbf{x}^k, \overline{\mathbf{h}}^k)$, $\forall\ k \in \{1, ..., p\}$*; entonces para cada* $i \in \{n+1, ..., n+p\}$ *tal que* $i = n+r$*, con* $r_i \in \{1, ..., p\}$ *se cumple que:* $v_{ij} = \alpha(1, X_j^{r_i})$ *y* $\lambda_{ij} = \alpha(0, \overline{X}_j^{r_i})\ \forall\ j \in \{1, ..., n+p\}$.

Demostración.- Por la forma de construir los vectores $\mathbf{X}^k = \tau^e(\mathbf{x}^k, \mathbf{h}^k)$ y $\overline{\mathbf{X}}^k = \tau^e(\mathbf{x}^k, \overline{\mathbf{h}}^k)$, se tiene que, $X_i^{r_i} = 1$ y $\overline{X}_i^{r_i} = 0$, además de que $X_i^{\mu} = 0$ y $\overline{X}_i^{\mu} = 1\ \forall \mu \neq r_i$ tal que $\mu \in \{1, ..., p\}$. Por lo anterior, y usando la definición de $\alpha$, $\alpha(X_i^{r_i}, X_j^{r_i}) = \alpha(1, X_j^{r_i})$ y $\alpha(X_i^{\mu}, X_j^{\mu}) = \alpha(0, X_j^{\mu})$ lo cual implica que, sin importar los valores de $X_j^{r_i}$ y $X_j^{\mu}$, se cumple $\alpha(X_i^{r_i}, X_j^{r_i}) \geq \alpha(X_i^{\mu}, X_j^{\mu})$ , de donde:

$$v_{ij} = \bigvee_{\mu=1}^{p} \alpha(X_i^{\mu}, X_j^{\mu}) = \alpha(X_i^{r_i}, X_j^{r_i}) = \alpha(1, X_j^{r_i})$$

También se tiene que $\alpha(\overline{X}_i^{r_i}, \overline{X}_j^{r_i}) = \alpha(0, \overline{X}_j^{r_i})$ y $\alpha(\overline{X}_i^{\mu}, \overline{X}_j^{\mu}) = \alpha(1, \overline{X}_j^{\mu})$, lo cual implica que, sin importar los valores de $\overline{X}_j^{r_i}$ y $\overline{X}_j^{\mu}$, se cumple $\alpha(\overline{X}_i^{r_i}, \overline{X}_j^{r_i}) \leq \alpha(\overline{X}_i^{\mu}, \overline{X}_j^{\mu})$, de donde:

$$\lambda_{ij} = \bigwedge_{\mu=1}^{p} \alpha(\overline{X}_i^{\mu}, \overline{X}_j^{\mu}) = \alpha(\overline{X}_i^{r_i}, \overline{X}_j^{r_i}) = \alpha(0, \overline{X}_j^{r_i})$$

$\mu \in \{1, ..., p\}, \forall\ j \in \{1, ..., n+p\}$.■

Ejemplo 4.15 (Ilustra el Lema 4.7) Tomemos los vectores $\mathbf{X}^k$ para $k = 1, 2, 3, 4$ de ejemplo 4.2

$$\mathbf{X}^1 = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}, \quad \mathbf{X}^2 = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \end{bmatrix}, \quad \mathbf{X}^3 = \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \end{bmatrix}, \quad \mathbf{X}^4 = \begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}, \qquad \mathbf{V} = \begin{bmatrix} 1 & 1 & 2 & 2 & 2 & 2 & 2 & 1 \\ 1 & 1 & 2 & 2 & 2 & 2 & 2 & 2 \\ 2 & 2 & 1 & 2 & 2 & 2 & 2 & 2 \\ 1 & 1 & 1 & 1 & 1 & 2 & 2 & 2 \\ 1 & 1 & 1 & 1 & 1 & 2 & 2 & 2 \\ 2 & 2 & 2 & 2 & 2 & 1 & 2 & 2 \\ 2 & 2 & 1 & 2 & 2 & 2 & 1 & 2 \\ 1 & 1 & 2 & 2 & 2 & 2 & 2 & 1 \end{bmatrix}$$

En este ejemplo $i \in \{5, 6, 7, 8\}$ y $j \in \{1, 2, ..., 8\}$. Debido a que estamos utilizando la memoria autoasociativa Alfa-Beta *max*, en la etapa de aprendizaje se debe tomar el valor máximo. Usando la definición de $\alpha$, el valor máximo que se puede alcanzar es 2 el cual se obtiene con $\alpha(1,0)=2$. Con un valor de cero el máximo que se puede obtener es $\alpha(0,0)=1$. Por lo tanto, la componente que establecerá un valor máximo será aquella que sea 1. La única componente, de los cuatro vectores para $i = 5$, que tiene el valor de 1 es la de $\mathbf{X}^1$. Esto es, $X_5^1 = 1$ establecerá un valor máximo, entonces $v_{5j} = \alpha(X_5^1, X_j^{r_i}) = \alpha(1, X_j^{r_i})$.

Para $i = 6$, $X_6^2 = 1$, entonces $v_{6j} = \alpha(X_6^2, X_j^{r_i}) = \alpha(1, X_j^{r_i})$.
Para $i = 7$, $X_7^3 = 1$, entonces $v_{7j} = \alpha(X_7^3, X_j^{r_i}) = \alpha(1, X_j^{r_i})$.
Para $i = 8$, $X_8^4 = 1$, entonces $v_{8j} = \alpha(X_8^4, X_j^{r_i}) = \alpha(1, X_j^{r_i})$.

Ahora, tomemos los vectores $\overline{\mathbf{X}}^k$ para $k = 1, 2, 3, 4$ de ejemplo 4.6

$$\overline{\mathbf{X}}^1 = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 0 \\ 1 \\ 1 \\ 1 \end{bmatrix}, \quad \overline{\mathbf{X}}^2 = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 1 \\ 1 \end{bmatrix}, \quad \overline{\mathbf{X}}^3 = \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \\ 1 \\ 1 \\ 0 \\ 1 \end{bmatrix}, \quad \overline{\mathbf{X}}^4 = \begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 1 \\ 1 \\ 1 \\ 0 \end{bmatrix}, \qquad \mathbf{\Lambda} = \begin{bmatrix} 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 \end{bmatrix}$$

En el caso de la memoria autoasociativa Alfa-Beta *min*, en la fase de aprendizaje debe tomarse el valor mínimo. El valor mínimo que se puede encontrar es 0 y se da cuando $\alpha$

(0,1)=0. Por lo tanto, la componente que establecerá un valor mínimo será aquella que sea 0. La única componente, de los cuatro vectores para $i = 5$, que tiene el valor de 0 es la de $\overline{\mathbf{X}}^1$. Esto es, $\overline{X}_5^1 = 0$ establecerá un valor mínimo, entonces $\lambda_{5j} = \alpha(\overline{X}_5^1, X_j^{r_i}) = \alpha(0, \overline{X}_j^{r_i})$.

Para $i = 6$, $\overline{X}_6^2 = 1$, entonces $\lambda_{6j} = \alpha(\overline{X}_6^2, X_j^{r_i}) = \alpha(0, \overline{X}_j^{r_i})$.
Para $i = 7$, $\overline{X}_7^3 = 1$, entonces $\lambda_{7j} = \alpha(\overline{X}_7^3, \overline{X}_j^{r_i}) = \alpha(0, \overline{X}_j^{r_i})$.
Para $i = 8$, $\overline{X}_8^4 = 1$, entonces $\lambda_{8j} = \alpha(\overline{X}_8^4, \overline{X}_j^{r_i}) = \alpha(0, \overline{X}_j^{r_i})$.

Corolario 4.3 *Sea* $\{(\mathbf{X}^k, \mathbf{X}^k) \mid k = 1, ..., p\}$ *el conjunto fundamental de una memoria autoasociativa Alfa-Beta max representada por* $\mathbf{V}$*, con* $\mathbf{X}^k = \tau^e(\mathbf{x}^k, \mathbf{h}^k)$, $\forall\ k \in \{1, ..., p\}$ *y sea* $\left\{\left(\overline{\mathbf{X}}^k, \overline{\mathbf{X}}^k\right) \mid k = 1,..., p\right\}$ *el conjunto fundamental de una memoria autoasociativa Alfa-Beta min representada por* $\mathbf{\Lambda}$*, con* $\overline{\mathbf{X}}^k = \tau^e(\mathbf{x}^k, \overline{\mathbf{h}}^k)$, $\forall\ k \in \{1, ..., p\}$*; entonces* $\nu_{ij} = \lambda_{ij} + 1$, $\forall\ i \in \{n+1, ..., n+p\}$*,* $i = n + r_i$*, con* $ri \in \{1, ..., p\}$ *y* $\forall\ j \in \{1, ..., n\}$.

Demostración.- Sean dos índices $i \in \{n+1, ..., n+p\}$ y $j \in \{1, ..., n\}$ seleccionados arbitrariamente. Por Lema 4.7, las expresiones para calcular las $ij$-ésimas componentes de las memorias $\mathbf{V}$ y $\mathbf{\Lambda}$ toman los siguientes valores:

$$\nu_{ij} = \alpha(1, X_j^{r_i})$$

y

$$\lambda_{ij} = \alpha(0, \overline{X}_j^{r_i})$$

Considerando que para $\forall\ j \in \{1, ..., n\}$ $X_j^{r_i} = \overline{X}_j^{r_i}$, hay dos casos posibles:

CASO 1 $X_j^{r_i} = 0 = \overline{X}_j^{r_i}$. Se tienen los siguientes valores: $\nu_{ij} = \alpha(1,0) = 2$ y $\lambda_{ij} = \alpha(0,0) = 1$, por lo que $\nu_{ij} = \lambda_{ij} + 1$.

CASO 2 $X_j^{r_i} = 1 = \overline{X}_j^{r_i}$. Se tienen los siguientes valores: $\nu_{ij} = \alpha(1,1) = 1$ y $\lambda_{ij} = \alpha(0,1) = 0$, por lo que $\nu_{ij} = \lambda_{ij} + 1$.

Dado que los índices $i$ y $j$ fueron escogidos arbitrariamente en sus respectivos dominios, el resultado $\nu_{ij} = \lambda_{ij} + 1$ es válido $\forall\ i \in \{n+1, ..., n+p\}$ y $\forall\ j \in \{1, ..., n\}$.■

Ejemplo 4.16 (Ilustra el Corolario 4.3) Para este ejemplo utilizaremos las memorias V y $\mathbf{\Lambda}$ mostradas en el ejemplo 4.15.

$$\mathbf{V} = \begin{bmatrix} 1 & 1 & 2 & 2 & 2 & 2 & 2 & 1 \\ 1 & 1 & 2 & 2 & 2 & 2 & 2 & 2 \\ 2 & 2 & 1 & 2 & 2 & 2 & 2 & 2 \\ 1 & 1 & 1 & 1 & 1 & 2 & 2 & 2 \\ 1 & 1 & 1 & 1 & 1 & 2 & 2 & 2 \\ 2 & 2 & 2 & 2 & 2 & 1 & 2 & 2 \\ 2 & 2 & 1 & 2 & 2 & 2 & 1 & 2 \\ 1 & 1 & 2 & 2 & 2 & 2 & 2 & 1 \end{bmatrix}, \qquad \mathbf{\Lambda} = \begin{bmatrix} 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 \end{bmatrix}$$

Con $n = 4$ y $p = 4$, $i \in \{5, 6, 7, 8\}$ y $j \in \{1, 2, 3, 4\}$

$\nu_{51} = \lambda_{51} + 1 = 0+1 = 1$ $\nu_{61} = \lambda_{61} + 1 = 1+1 = 2$ $\nu_{71} = \lambda_{71} + 1 = 1+1 = 2$ $\nu_{81} = \lambda_{81} + 1 = 0+1 = 1$

$\nu_{52} = \lambda_{52} + 1 = 0+1 = 1$ $\nu_{62} = \lambda_{62} + 1 = 1+1 = 2$ $\nu_{72} = \lambda_{72} + 1 = 1+1 = 2$ $\nu_{82} = \lambda_{82} + 1 = 0+1 = 1$

$\nu_{53} = \lambda_{53} + 1 = 0+1 = 1$ $\nu_{63} = \lambda_{63} + 1 = 1+1 = 2$ $\nu_{73} = \lambda_{73} + 1 = 0+1 = 1$ $\nu_{83} = \lambda_{83} + 1 = 1+1 = 2$

$\nu_{54} = \lambda_{54} + 1 = 0+1 = 1$ $\nu_{64} = \lambda_{64} + 1 = 1+1 = 2$ $\nu_{74} = \lambda_{74} + 1 = 1+1 = 2$ $\nu_{84} = \lambda_{84} + 1 = 1+1 = 2$

Lema 4.8 *Sea* $\{(\mathbf{X}^k, \mathbf{X}^k) \mid k = 1, ..., p\}$ *el conjunto fundamental de una memoria autoasociativa Alfa-Beta max representada por* $\mathbf{V}$*, con* $\mathbf{X}^k = \tau^e(\mathbf{x}^k, \mathbf{h}^k)\ \forall\ k \in \{1, ..., p\}$*, y sea* $\left\{\left(\overline{\mathbf{X}}^k, \overline{\mathbf{X}}^k\right) \mid k = 1,..., p\right\}$ *el conjunto fundamental de una memoria autoasociativa Alfa-Beta min representada por* $\mathbf{\Lambda}$*, con* $\overline{\mathbf{X}}^k = \tau^e(\mathbf{x}^k, \overline{\mathbf{h}}^k)\ \forall\ k \in \{1, ..., p\}$*. Además, si se define el vector* $\mathbf{u} \in A^p$ *como* $\mathbf{u} = \sum_{i=1}^{p} \mathbf{h}^i$*, y se toma un índice fijo* $\forall\ r \in \{1, ..., p\}$*, consideremos dos versiones ruidosas del patrón* $X^r \in A^{n+p}$*: el vector* $\mathbf{F} = \tau^e(\mathbf{x}^r, \mathbf{u}) \in A^{n+p}$ *que es una versión alterada con ruido aditivo del patrón* $X^r$*, y el vector* $\mathbf{G} = \tau^e(\mathbf{x}^r, \mathbf{w}) \in A^{n+p}$*, el cual es una versión alterada con ruido sustractivo del patrón* $\overline{\mathbf{X}}^r$*, siendo* $\mathbf{w} \in A^p$ *un vector cuyas componentes toman los valores* $w_i = u_i - 1\ \forall\ i \in \{1, ..., p\}$*. Si en la fase de recuperación se presenta* $\mathbf{G}$ *a la entrada de la memoria* $\mathbf{\Lambda}$ *y se presenta* $\mathbf{F}$ *a la entrada de la memoria* $\mathbf{V}$*, y si además se cumple* $(\mathbf{\Lambda} \nabla_\beta \mathbf{G})_t = 0$ *para un índice* $t \in \{n+1, ..., n+p\}$ *fijo tal que* $t \neq n + r$*, entonces* $(\mathbf{V} \Delta_\beta \mathbf{F})_t = 0$.

Demostración.- Por la forma de construir los vectores $\mathbf{X}^r$, $\mathbf{F}$ y $\mathbf{G}$, se tiene que $F_t = 1$ es la componente del vector con ruido aditivo que le corresponde a la componente $X_t^r$, y $G_t = 0$ es la componente del vector con ruido sustractivo que le corresponde a la componente $\overline{X}_t^r$; además, dado que $t \neq n + r$, se puede ver que $X_t^r \neq 1$, es decir $X_t^r = 0$, y $\overline{X}_t^r = 1$. Hay dos casos posibles:

CASO 1 El patrón $\mathbf{F}$ no contiene componentes con valor 0; es decir $F_j = 1\ \forall\ j \in \{1, ...,n+ p\}$. Por Lema 4.3 $\nu_{tj} \neq 0\ \forall\ j \in \{1, ...,n+ p\}$, por lo que $\beta(\nu_{tj}, F_j)\ \forall\ j \in \{1, ...,n+ p\}$, lo cual

significa que $\left(\mathbf{V}\Delta_{\beta}\mathbf{F}\right)_t = \wedge_{j=1}^{n+p}\beta(v_{tj}, F_j) = 1$. Es decir, la expresión ($\mathbf{V}\ \Delta_{\beta}\ \mathbf{F}$)$_t$ = 0 es falsa. La única posibilidad de que el teorema se cumpla, es que la expresión ($\mathbf{\Lambda}\ \nabla_{\beta}\ \mathbf{G}$)$_t$ = 0 sea falsa también; es decir, debemos mostrar que ($\mathbf{\Lambda}\ \nabla_{\beta}\ \mathbf{G}$)$_t$ = 1. De acuerdo con el Corolario 4.2, lo anterior se cumple si para cada $t \in \{n+1, ..., n+p\}$ con $t \neq n + r$, existe $j \in \{1, ...,n+ p\}$ tal que ($G_j = 1$ AND $\lambda_{tj} \neq 0$). Ahora, $t \neq n + r$ indica que $\exists\ s \in \{1, ..., p\}$, $s \neq r$ tal que $t = n + s$, y por Lema 4.7 $\alpha(\overline{X}_t^s, \overline{X}_j^s) \leq \alpha(\overline{X}_t^{\mu}, \overline{X}_j^{\mu})\ \forall\ \mu \in \{1, ..., p\}$, $\forall\ j \in \{1, ...,n+ p\}$, de donde se obtiene que $\lambda_{tj} = \wedge_{j=1}^{p}\alpha(\overline{X}_t^{\mu}, \overline{X}_j^{\mu}) = \alpha(\overline{X}_t^s, \overline{X}_j^s)$, y al notar la igualdad $\overline{X}_t^s = \overline{X}_{n+s}^s = 0$, se cumple que:

$$\lambda_{tj} = \alpha(0, \overline{X}_j^s)\ \ \forall\ j \in \{1, ...,n+ p\}$$

Por otro lado, $\forall\ i \in \{1, ..., n\}$ se cumplen las siguientes igualdades: $\overline{X}_i^r = x_i^r = 1$ y $\overline{X}_i^s = x_i^s$ y además, tomando en cuenta que $\mathbf{x}^r \neq \mathbf{x}^s$, es claro que $\exists\ h \in \{1, ..., p\}$ tal que $x_h^s \neq x_h^r$; es decir $x_h^s = 0 = X_h^s$ y por ello:

$$\lambda_{th} = \alpha(0,0) = 1$$

Finalmente, dado que $\forall\ i \in \{1, ..., n\}$ se cumple que $G_i = \overline{X}_i^r = x_i^r = 1$, en particular $G_h = 1$. Entonces hemos demostrado que para cada $t \in \{n+1, ..., n+p\}$ con $t \neq n + r$, existe $j \in \{1, ...,n+ p\}$ tal que ($G_j = 1$ AND $\lambda_{tj} \neq 0$), y por Corolario 4.2 se cumple que ($\mathbf{\Lambda}\ \nabla_{\beta}\ \mathbf{G}$)$_t$ = 1, por lo que la expresión ($\mathbf{\Lambda}\ \nabla_{\beta}\ \mathbf{G}$)$_t$ = 1 es falsa.

CASO 2 El patrón **F** contiene, además de las componentes con valor 1, al menos una componente con valor 0; es decir $\exists\ h \in \{1, ..., n+p\}$ tal que $F_h = 0$. Por la forma de construir los vectores **G** y **F** $\forall\ i \in \{1, ..., n\}$ $G_i = F_i$ y, además, necesariamente $1 \leq h \leq n$ y por ello $F_h = G_h = 0$. Por hipótesis $\exists\ t \in \{n+1, ..., n+p\}$ fijo tal que $t \neq n + r$ y ($\mathbf{\Lambda}\ \nabla_{\beta}\ \mathbf{G}$)$_t$ = 0 , y por Lema 4.6 $\forall\ j \in \{1, ..., n+p\}$ ($G_j = 1 \rightarrow \lambda_{tj} = 0$). Por la forma de construir el vector **G**, se tiene que $\forall\ j \in \{n+1, ..., n+p\}$ $G_j = 0$, por lo que la expresión anterior queda así: $\forall\ j \in \{1, ...,n\}$ ($G_j = 1 \rightarrow \lambda_{tj} = 0$). Sea un conjunto $J$, subconjunto propio de $\{1, ...,n\}$, definido así: $J = \{j \in \{1, ...,n\} \mid G_j = 1\}$; el hecho de que $J$ es subconjunto propio de $\{1, ...,n\}$ está garantizado por la existencia de $G_h = 0$. Ahora, $t \neq n + r$ indica que $\exists\ s \in \{1, ..., p\}$, $s \neq r$ tal que $t = n + s$, y por Lema 4.7 $v_{tj} = \alpha(1, X_j^s)$ y $\lambda_{tj} = \alpha(0, \overline{X}_j^s)\ \forall\ j \in \{1, ...,n+ p\}$, de donde se obtiene que $\forall\ j \in J$, $\overline{X}_j^s = 1$ porque si no fuese así, $\lambda_{tj} \neq 0$. Es decir, para cada $j \in J$ $\overline{X}_j^s = 1 = G_j$ y esto significa que los patrones $\mathbf{X}^r$ y $\mathbf{X}^s$ coinciden con valor 1 en todas las componentes con índice $j \in J$. Consideremos ahora el complemento del conjunto $J$, el cual se define como $J^c = \{j \in \{1, ...,n\} \mid G_j = 0\}$ la existencia de al menos un valor $j_0 \in J^c$ para el cual $G_{j_0} = 0$ y $\overline{X}_{j_0}^s = 1$ está garantizado por el hecho conocido de que $\mathbf{x}^r \neq \mathbf{x}^s$; veamos, si $\overline{X}_j^s = 0\ \forall\ j \in J^c$ entonces $\forall\ j \in \{1, ...,n\}$ se cumple que $\overline{X}_j^s = G_j$, lo cual significaría que $\mathbf{x}^r = \mathbf{x}^s$. Dado que $\exists\ j_0 \in J^c$ para el cual $G_{j_0} = 0$ y $\overline{X}_{j_0}^s = 1$, eso significa que $\exists\ j_0 \in J^c$ para el cual $F_{j_0} = 0$ y $X_{j_0}^s = 1$. Ahora $\beta(v_{tj_0}, F_{j_0}) = \beta(\alpha(1, X_{j_0}^s), 0) = \beta(\alpha(1,1), 0) = \beta(1,0) = 0$ y

finalmente

$$\left(\mathbf{V}\Delta_\beta\mathbf{F}\right)_t = \bigwedge_{j=1}^{n+p}\beta(v_{tj},F_j) = \beta(v_{tj_0},F_{j_0}) = 0\,.\ \blacksquare$$

Ejemplo 4.17 (Ilustra el Lema 4.8) Del ejemplo 4.2 tomemos $\mathbf{X}^2$ y construyamos el vector ruidoso **F**, con ruido aditivo, del ejemplo 4.6 tomemos $\overline{\mathbf{X}}^2$ y construimos el vector ruidoso **G**, con ruido sustractivo, y utilizaremos las memorias V y $\mathbf{\Lambda}$ mostradas en el ejemplo 4.15.

$$\mathbf{X}^2 = \begin{bmatrix}0\\0\\0\\0\\0\\1\\0\\0\end{bmatrix},\ \mathbf{F} = \begin{bmatrix}0\\0\\0\\0\\1\\1\\1\\1\end{bmatrix},\ \overline{\mathbf{X}}^2 = \begin{bmatrix}0\\0\\0\\0\\1\\0\\1\\1\end{bmatrix},\ \mathbf{G} = \begin{bmatrix}0\\0\\0\\0\\0\\0\\0\\0\end{bmatrix},\ \mathbf{V} = \begin{bmatrix}1&1&2&2&2&2&2&1\\1&1&2&2&2&2&2&2\\2&2&1&2&2&2&2&2\\1&1&1&1&1&2&2&2\\1&1&1&1&1&2&2&2\\2&2&2&2&2&1&2&2\\2&2&1&2&2&2&1&2\\1&1&2&2&2&2&2&1\end{bmatrix},\ \mathbf{\Lambda} = \begin{bmatrix}1&1&0&1&0&0&0&0\\1&1&0&1&0&0&0&0\\0&0&1&1&0&0&0&0\\0&0&0&1&0&0&0&0\\0&0&0&0&1&0&0&0\\1&1&1&1&0&1&0&0\\1&1&0&1&0&0&1&0\\0&0&1&1&0&0&0&1\end{bmatrix}$$

Ahora le presentamos **F** a **V** y **G** a $\mathbf{\Lambda}$ para obtener

$$\mathbf{V}\Delta_\beta\mathbf{F} = \begin{bmatrix}0\\0\\0\\0\\0\\1\\0\\0\end{bmatrix},\qquad \mathbf{\Lambda}\nabla_\beta\mathbf{G} = \begin{bmatrix}0\\0\\0\\0\\0\\0\\0\\0\end{bmatrix}$$

Con $n = 4$ y $p = 4$, $t \in \{5, 6, 7, 8\}$ y para este ejemplo $r = 2$, por tanto $t \neq 4+2 = 6$. Entonces,

para $t = 5$ $(\mathbf{\Lambda}\nabla_\beta\mathbf{G})_5 = 0$, $(\mathbf{V}\Delta_\beta\mathbf{F})_5 = 0$.
para $t = 7$ $(\mathbf{\Lambda}\nabla_\beta\mathbf{G})_7 = 0$, $(\mathbf{V}\Delta_\beta\mathbf{F})_7 = 0$.
para $t = 8$ $(\mathbf{\Lambda}\nabla_\beta\mathbf{G})_8 = 0$, $(\mathbf{V}\Delta_\beta\mathbf{F})_8 = 0$.

Lema 4.9 *Sea* $\{(\mathbf{X}^k, \mathbf{X}^k) \mid k = 1, ..., p\}$ *el conjunto fundamental de una memoria autoasociativa Alfa-Beta max representada por* $\mathbf{V}$*, con* $\mathbf{X}^k = \tau^e(\mathbf{x}^k, \mathbf{h}^k)\ \forall\ k \in \{1, ..., p\}$*, y sea* $\left\{\left(\overline{\mathbf{X}}^k, \overline{\mathbf{X}}^k\right) \mid k = 1,..., p\right\}$ *el conjunto fundamental de una memoria autoasociativa Alfa-Beta min representada por* $\mathbf{\Lambda}$*, con* $\overline{\mathbf{X}}^k = \tau^e(\mathbf{x}^k, \overline{\mathbf{h}}^k)\ \forall\ k \in \{1, ..., p\}$*. Además, si se define el vector* $\mathbf{u} \in A^p$ *como* $\mathbf{u} = \sum_{i=1}^{p} \mathbf{h}^i$*, y se toma un índice fijo* $\forall\ r \in \{1, ..., p\}$*, consideremos dos versiones ruidosas del patrón* $X^r \in A^{n+p}$*: el vector* $\mathbf{F} = \tau^e(\mathbf{x}^r, \mathbf{u}) \in A^{n+p}$ *que es una versión alterada con ruido aditivo del patrón* $X^r$*, y el vector* $\mathbf{G} = \tau^e(\mathbf{x}^r, \mathbf{w}) \in A^{n+p}$*, el cual es una versión alterada con ruido sustractivo del patrón* $\overline{\mathbf{X}}^r$*, siendo* $\mathbf{w} \in A^p$ *un vector cuyas componentes toman los valores* $w_i = u_i - 1\ \forall\ i \in \{1, ..., p\}$*. Si en la fase de recuperación se presenta* $\mathbf{F}$ *a la entrada de la memoria* $\mathbf{V}$ *y se presenta* $\mathbf{G}$ *a la entrada de la memoria* $\mathbf{\Lambda}$*, y si además se cumple* $(\mathbf{V} \Delta_\beta \mathbf{F})_t = 1$ *para un índice* $t \in \{n+1, ..., n+p\}$ *fijo tal que* $t \neq n + r$*, entonces* $(\mathbf{\Lambda} \nabla_\beta \mathbf{G})_t = 1$.

Demostración Por la forma de construir los vectores $\mathbf{X}^r$, $\mathbf{F}$ y $\mathbf{G}$, se tiene que $F_t = 1$ es la componente del vector con ruido aditivo que le corresponde a la componente $X_t^r$, y $G_t = 0$ es la componente del vector con ruido sustractivo que le corresponde a la componente $\overline{X}_t^r$; además, dado que $t \neq n + r$, se puede ver que $X_t^r \neq 1$, es decir $X_t^r = 0$, y $\overline{X}_t^r = 1$. Hay dos casos posibles:

CASO 1 El patrón $\mathbf{G}$ no contiene componentes con valor 1; es decir $G_j = 0\ \forall\ j \in \{1, ...,n+ p\}$. Por el Lema 4.5 $\lambda_{tj} \neq 2\ \forall\ j \in \{1, ..., n+p\}$, por lo que $\beta(\lambda_{tj}, G_j) = 0\ \forall\ j \in \{1, ..., n+p\}$, lo cual significa que $\left(\mathbf{\Lambda} \nabla_\beta \mathbf{G}\right)_t = \vee_{j=1}^{n+p} \beta(\lambda_{tj}, G_j) = 0$. Es decir, la expresión $(\mathbf{\Lambda} \nabla_\beta \mathbf{G})_t = 1$ es falsa. La única posibilidad de que el teorema se cumpla, es que la expresión $(\mathbf{V} \Delta_\beta \mathbf{F})_t = 1$ sea falsa también; es decir, debemos mostrar que $(\mathbf{V} \Delta_\beta \mathbf{F})_t = 0$. De acuerdo con el Corolario 4.1, lo anterior se cumple si para cada $t \in \{n+1, ..., n+p\}$ con $t \neq n + r$, existe $j \in \{1, ..., n+p\}$ tal que ($F_j = 0$ AND $\nu_{tj} \neq 2$). Ahora sea $t \neq n + r$, lo cual indica que $\exists\ s \in \{1, ..., p\}$, $s \neq r$ tal que $t = n + s$, y por el Lema 4.6 $\alpha(X_t^s, X_j^s) \geq \alpha(X_t^\mu, X_j^\mu)\ \forall\ \mu \in \{1, ..., p\}$, $\forall\ j \in \{1, ...,n+ p\}$, de donde se obtiene que $\nu_{tj} = \vee_{\mu=1}^{p} \alpha(X_t^\mu, X_j^\mu) = \alpha(X_t^s, X_j^s)$, y al notar la igualdad $X_t^s = X_{n+s}^s = 1$, se cumple que:

$$\nu_{tj} = \alpha(1, X_j^s)\ \forall\ j \in \{1, ...,n+ p\}$$

Por otro lado, $\forall\ i \in \{1, ..., n\}$ se cumplen las siguientes igualdades: $X_i^r = x_i^r = 0$ y $X_i^s = x_i^s$ y además, tomando en cuenta que $\mathbf{x}^r \neq \mathbf{x}^s$, es claro que $\exists\ h \in \{1, ..., p\}$ tal que $x_h^s \neq x_h^r$; es decir $x_h^s = 1 = X_h^s$ y por ello:

$$\nu_{th} = \alpha(1, X_h^s) = \alpha(1,1) = 1$$

Finalmente, dado que $\forall\ i \in \{1, ..., n\}$ se cumple que $F_i = X_i^r = x_i^r = 0$, en particular $F_h = 0$. Entonces hemos demostrado que para cada $t \in \{n+1, ..., n+p\}$ con $t \neq n + r$, existe $j \in \{1, ...,n+ p\}$ tal que ($F_j = 0$ AND $\nu_{tj} \neq 2$), y por Corolario 4.1 se cumple que $(\mathbf{V} \Delta_\beta \mathbf{F})_t = 0$, por lo que la expresión $(\mathbf{V} \Delta_\beta \mathbf{F})_t = 1$ es falsa.

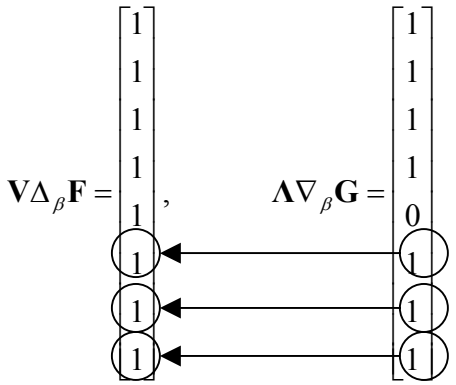
CASO 2 El patrón **G** contiene, además de las componentes con valor 0, al menos una componente con valor 1; es decir $\exists\ h \in \{1, ..., n+p\}$ tal que $G_h = 1$. Por la forma de construir los vectores **G** y **F** $\forall\ i \in \{1, ..., n\}\ G_i = F_i$ y, además, necesariamente $1 \le h \le n$ y por ello $F_h = G_h = 0$. Por hipótesis $\exists\ t \in \{n+1, ..., n+p\}$ fijo tal que $t \ne n + r$ y $(\mathbf{V} \Delta_\beta \mathbf{F})_t = 1$, y por el Lema 4.4 $\forall\ j \in \{1, ...,n+ p\}$ $(F_j = 0 \rightarrow v_{tj} = 2)$. Por la forma de construir el vector **F**, se tiene que $\forall\ j \in \{n+1, ..., n+p\}\ G_j = 1$, por lo que la expresión anterior queda así: $\forall\ j \in \{1, ...,n+ p\}$ $(F_j = 0 \rightarrow v_{tj} = 2)$. Sea un conjunto $J$, subconjunto propio de $\{1, ...,n\}$, definido así: $J = \{j \in \{1, ...,n\} \mid F_j = 0\}$; el hecho de que $J$ es subconjunto propio de $\{1, ...,n\}$ está garantizado por la existencia de $G_h = 1$. Ahora, $t \ne n + r$ indica que $\exists\ s \in \{1, ..., p\}$, $s \ne r$ tal que $t = n + s$, y por el Lema 4.7 $v_{tj} = \alpha(1, X_j^s)$ y $\lambda_{tj} = \alpha(0, \overline{X}_j^s)\ \forall\ j \in \{1, ...,n+ p\}$, de donde se obtiene que $\forall\ j \in J,\ X_j^s = 0$ porque si no fuese así, $v_{tj} \ne 0$. Es decir, para cada $j \in J\ \ X_j^s = 0 = F_j$ y esto significa que los patrones $\mathbf{X}^r$ y $\mathbf{X}^s$ coinciden con valor 0 en todas las componentes con índice $j \in J$. Consideremos ahora el complemento del conjunto $J$, el cual se define como $J^c = \{j \in \{1, ...,n\} \mid F_j = 1\}$ la existencia de al menos un valor $j_0 \in J^c$ para el cual $F_{j_0} = 1$ y $X_{j_0}^s = 0$ está garantizado por el hecho conocido de que $\mathbf{x}^r \ne \mathbf{x}^s$; veamos, si $X_j^s = 1\ \forall\ j \in J^c$ entonces $\forall\ j \in \{1, ...,n\}$ se cumple que $X_j^s = F_j$, lo cual significaría que $\mathbf{x}^r = \mathbf{x}^s$. Dado que $\exists\ j_0 \in J^c$ para el cual $F_{j_0} = 1$ y $X_{j_0}^s = 0$, eso significa que $\exists\ j_0 \in J^c$ para el cual $G_{j_0} = 1$ y $\overline{X}_{j_0}^s = 0$. Ahora $\beta(\lambda_{tj_0}, G_{j_0}) = \beta(\alpha(0, \overline{X}_{j_0}^s), 1) = \beta(\alpha(0,0),1) = \beta(1,1) = 1$ y finalmente

$$\left(\mathbf{\Lambda} \nabla_\beta \mathbf{G}\right)_t = \bigvee_{j=1}^{n+p} \beta(\lambda_{tj}, G_j) = \beta(\lambda_{tj_0}, G_{j_0}) = 1\,.\ \blacksquare$$

Ejemplo 4.18 (Ilustra el Lema 4.9) Del ejemplo 4.2 tomemos $\mathbf{X}^1$ y construyamos el vector ruidoso **F**, con ruido aditivo, del ejemplo 4.6 tomemos $\overline{\mathbf{X}}^1$ y construimos el vector ruidoso **G**, con ruido sustractivo, y utilizaremos las memorias V y **Λ** mostradas en el ejemplo 4.15.

$$\mathbf{X}^1 = \begin{bmatrix}1\\1\\1\\1\\1\\0\\0\\0\end{bmatrix},\ \mathbf{F} = \begin{bmatrix}1\\1\\1\\1\\1\\1\\1\\1\end{bmatrix},\ \overline{\mathbf{X}}^1 = \begin{bmatrix}1\\1\\1\\1\\0\\1\\1\\1\end{bmatrix},\ \mathbf{G} = \begin{bmatrix}1\\1\\1\\1\\0\\0\\0\\0\end{bmatrix},\ \mathbf{V} = \begin{bmatrix}1&1&2&2&2&2&2&1\\1&1&2&2&2&2&2&2\\2&2&1&2&2&2&2&2\\1&1&1&1&1&2&2&2\\1&1&1&1&1&2&2&2\\2&2&2&2&2&1&2&2\\2&2&1&2&2&2&1&2\\1&1&2&2&2&2&2&1\end{bmatrix},\ \mathbf{\Lambda} = \begin{bmatrix}1&1&0&1&0&0&0&0\\1&1&0&1&0&0&0&0\\0&0&1&1&0&0&0&0\\0&0&0&1&0&0&0&0\\0&0&0&0&1&0&0&0\\1&1&1&1&0&1&0&0\\1&1&0&1&0&0&1&0\\0&0&1&1&0&0&0&1\end{bmatrix}$$

Ahora le presentamos **F** a **V** y **G** a **Λ** para obtener




Con $n = 4$ y $p = 4$, $t \in \{5, 6, 7, 8\}$ y para este ejemplo $r = 1$, por tanto $t \neq 4+1 = 5$. Entonces,

para $t = 6$ $(\mathbf{\Lambda} \nabla_\beta \mathbf{G})_6 = 1$, $(\mathbf{V} \Delta_\beta \mathbf{F})_6 = 1$.
para $t = 7$ $(\mathbf{\Lambda} \nabla_\beta \mathbf{G})_7 = 1$, $(\mathbf{V} \Delta_\beta \mathbf{F})_7 = 1$.
para $t = 8$ $(\mathbf{\Lambda} \nabla_\beta \mathbf{G})_8 = 1$, $(\mathbf{V} \Delta_\beta \mathbf{F})_8 = 1$.

Teorema 4.5 (Teorema Principal) *Sea* $\{(\mathbf{X}^k, \mathbf{X}^k) \mid k = 1, ..., p\}$ *el conjunto fundamental de una memoria autoasociativa Alfa-Beta max representada por* $\mathbf{V}$*, con* $\mathbf{X}^k = \tau^e(\mathbf{x}^k, \mathbf{h}^k)$ $\forall$ $k \in \{1, ..., p\}$*, y sea* $\left\{\left(\overline{\mathbf{X}}^k, \overline{\mathbf{X}}^k\right) \mid k = 1,..., p\right\}$ *el conjunto fundamental de una memoria autoasociativa Alfa-Beta min representada por* $\mathbf{\Lambda}$*, con* $\overline{\mathbf{X}}^k = \tau^e(\mathbf{x}^k, \overline{\mathbf{h}}^k)$ $\forall$ $k \in \{1, ..., p\}$*. Además, si se define el vector* $\mathbf{u} \in A^p$ *como* $\mathbf{u} = \sum_{i=1}^{p} \mathbf{h}^i$*, y se toma un índice fijo* $r \in \{1, ..., p\}$*, consideremos dos versiones ruidosas del patrón* $X^r \in A^{n+p}$*: el vector* $\mathbf{F} = \tau^e(\mathbf{x}^r, \mathbf{u}) \in A^{n+p}$ *que es una versión alterada con ruido aditivo del patrón* $X^r$*, y el vector* $\mathbf{G} = \tau^e(\mathbf{x}^r, \mathbf{w}) \in A^{n+p}$*, el cual es una versión alterada con ruido sustractivo del patrón* $\overline{\mathbf{X}}^r$*, siendo* $\mathbf{w} \in A^p$ *un vector cuyas componentes toman los valores* $w_i = u_i - 1$ $\forall$ $i \in \{1, ..., p\}$*. Asumamos que en la fase de recuperación se presenta* $\mathbf{G}$ *a la entrada de la memoria* $\mathbf{\Lambda}$ *y se presenta* $\mathbf{F}$ *a la entrada de la memoria* $\mathbf{V}$*, y se obtienen los patrones* $\mathrm{S} = \mathbf{\Lambda}\nabla_\beta \mathbf{G} \in A^{n+p}$ *y* $\mathbf{R} = \mathbf{V}\Delta_\beta\mathbf{F} \in A^{n+p}$*. Si al tomar como argumento el vector* $\mathbf{R}$ *se realiza la transformada vectorial de contracción* $\mathbf{r} = \tau^c(\mathbf{R}, n) \in A^p$*, y al tomar como argumento el vector* $\mathbf{S}$ *se realiza la transformada vectorial de contracción* $\mathbf{s} = \tau^c(\mathbf{S}, n) \in A^p$*, entonces* $\mathbf{H} = (\mathbf{r}$ AND $\overline{\mathbf{s}})$ *será el k-ésimo vector one-hot de p bits, donde* $\overline{\mathbf{s}}$ *es el vector negado de* $\mathbf{s}$*.*

Demostración.- Por la definición de transformada vectorial de contracción se tiene que $\mathbf{r}_i = \mathbf{R}_{i+n} = (\mathbf{V}\Delta_\beta\mathbf{F})_{i+n}$ y $s_i = S_{i+n} = (\mathbf{\Lambda}\nabla_\beta \mathbf{G})_{i+n}$ para $1 \leq i \leq p$; en particular, haciendo $i = k$ se tiene que $r_k = R_{k+n} = (\mathbf{V}\Delta_\beta\mathbf{F})_{k+n}$ y $s_k = S_{k+n} = (\mathbf{\Lambda}\nabla_\beta \mathbf{G})_{k+n}$. Por los Lemas 4.1 y 4.2 resulta que $\left(\mathbf{V}\Delta_\beta\mathbf{F}\right)_{n+k} = X^k_{n+k} = 1$ y $\left(\mathbf{\Lambda}\nabla_\beta\mathbf{G}\right)_{n+k} = \overline{X}^k_{n+k} = 0$; y por lo tanto:

$$\mathrm{H}_k = r_k \text{ AND } \overline{s}_k = 1 \text{ AND } \neg 0 = 1 \text{ AND } 1 = 1$$

Ahora, por el Lema 4.8 sabemos que si $(\mathbf{\Lambda} \nabla_\beta \mathbf{G})_t = 0$ tal que $t = i + n$ es un índice fijo con $t \neq n + k$, entonces $(\mathbf{V} \Delta_\beta \mathbf{F})_t = 0$; por lo tanto:

$$H_i = r_i \text{ AND } \bar{s}_i = (\mathbf{V} \Delta_\beta \mathbf{F})_t \text{ AND } \neg(\mathbf{\Lambda} \nabla_\beta \mathbf{G})_t = 0 \text{ AND } \neg 0 = 0 \text{ AND } 1 = 0$$

Por otro lado, por el Lema 4.9 se sabe que si $(\mathbf{V} \Delta_\beta \mathbf{F})_q = 1$ para un índice $q = i + n$ fijo tal que $q \neq n + k$, entonces $(\mathbf{\Lambda} \nabla_\beta \mathbf{G})_q = 1$; de acuerdo a lo anterior:

$$H_i = r_i \text{ AND } \neg s_i = (\mathbf{V} \Delta_\beta \mathbf{F})_q \text{ AND } \neg(\mathbf{\Lambda} \nabla_\beta \mathbf{G})_q = 1 \text{ AND } \neg 1 = 1 \text{ AND } 0 = 0$$

Entonces $H_i = 1$ para $i = k$ y $Hi = 0$ para $i \neq k$, por lo tanto, y de acuerdo con la Definición 1, H será el *k*-ésimo vector *one-hot* de *p* bits. ■

## 4.3 Fundamento Teórico de las Etapas 2 y 4

En esta sección se presenta el fundamento teórico que sustenta el diseño y operación de las Etapas 2 y 4, cuyo elemento principal es una variación original del *Linear Associator*. Esta modificación es otra contribución importante de este trabajo de tesis.

Sea $\{(\mathbf{x}^\mu, \mathbf{y}^\mu) \mid \mu = 1, 2, ..., p\}$ con $A = \{0, 1\}$, $\mathbf{x}^\mu \in A^n$ y $\mathbf{y}^\mu \in A^m$ el conjunto fundamental del *Linear Associator*.

La fase de Aprendizaje consiste de dos etapas:

- Para cada una de las *p* asociaciones $(\mathbf{x}^\mu, \mathbf{y}^\mu)$ se encuentra la matriz $\mathbf{y}^\mu \cdot (\mathbf{x}^\mu)^t$ de dimensiones *m* x *n*
- Se suman la *p* matrices para obtener la memoria

$$\mathbf{M} = \sum_{\mu=1}^{p} \mathbf{y}^\mu \cdot (\mathbf{x}^\mu)^t = [m_{ij}]_{m \times n}$$

de manera que la *ij*-ésima componente de la memoria M se expresa así:

$$m_{ij} = \sum_{\mu=1}^{p} y_i^\mu x_j^\mu$$

La fase de recuperación consiste en presentarle a la memoria un patrón de entrada $\mathbf{x}^\omega$, donde $\omega \in \{1, 2, ..., p\}$ y realizar la operación

$$\mathbf{M} \cdot \mathbf{x}^\omega = \left[ \sum_{\mu=1}^{p} \mathbf{y}^\mu \cdot (\mathbf{x}^\mu)^t \right] \cdot \mathbf{x}^\omega$$

La siguiente forma de expresión nos permite investigar las condiciones que se deben cumplir para que el método de recuperación propuesto dé como resultado salidas correctas.

$$\mathbf{M} \cdot \mathbf{x}^{\omega} = \mathbf{y}^{\omega} \cdot \left[ \left( \mathbf{x}^{\omega} \right)^{t} \cdot \mathbf{x}^{\omega} \right] + \sum_{\mu \neq \omega} \mathbf{y}^{\mu} \cdot \left[ \left( \mathbf{x}^{\mu} \right)^{t} \cdot \mathbf{x}^{\omega} \right]$$

Para que la expresión anterior arroje como resultado al patrón $\mathbf{y}^{\omega}$, es preciso que se cumplan dos igualdades:

- $\left[ \left( \mathbf{x}^{\omega} \right)^{t} \cdot \mathbf{x}^{\omega} \right] = 1$
- $\left[ \left( \mathbf{x}^{\mu} \right)^{t} \cdot \mathbf{x}^{\omega} \right] = 0$ siempre que $\mu \neq \omega$

Lo que nos indica que para que haya recuperación correcta, los vectores $\mathbf{x}^{\mu}$ deben ser ortonormales. Si sucede este caso, entonces, para $\mu = 1, 2, ..., p$, tenemos que:

$$\mathbf{y}^{1} \cdot (\mathbf{x}^{1})^{t} = \begin{pmatrix} y_1^1 \\ y_2^1 \\ \vdots \\ y_m^1 \end{pmatrix} \cdot \left( x_1^1, x_2^1, ..., x_n^1 \right) = \begin{pmatrix} y_1^1 & 0 & 0 & \cdots & 0_{(n)} \\ y_2^1 & 0 & 0 & \cdots & 0_{(n)} \\ \vdots & \vdots & \vdots & \cdots & \vdots \\ y_m^1 & 0 & 0 & \cdots & 0_{(n)} \end{pmatrix}$$

$$\mathbf{y}^{2} \cdot (\mathbf{x}^{2})^{t} = \begin{pmatrix} y_1^2 \\ y_2^2 \\ \vdots \\ y_m^2 \end{pmatrix} \cdot \left( x_1^2, x_2^2, ..., x_n^2 \right) = \begin{pmatrix} 0 & y_1^2 & 0 & \cdots & 0_{(n)} \\ 0 & y_2^2 & 0 & \cdots & 0_{(n)} \\ \vdots & \vdots & \vdots & \cdots & \vdots \\ 0 & y_m^2 & 0 & \cdots & 0_{(n)} \end{pmatrix}$$

$$\mathbf{y}^{p} \cdot (\mathbf{x}^{p})^{t} = \begin{pmatrix} y_1^p \\ y_2^p \\ \vdots \\ y_m^p \end{pmatrix} \cdot \left( x_1^p, x_2^p, ..., x_n^p \right) = \begin{pmatrix} 0 & 0 & 0 & \cdots & y_{1\,(n)}^p \\ 0 & 0 & 0 & \cdots & y_{2\,(n)}^p \\ \vdots & \vdots & \vdots & \cdots & \vdots \\ 0 & 0 & 0 & \cdots & y_{m(n)}^p \end{pmatrix}$$

Por lo tanto,

$$\mathbf{M} = \sum_{\mu=1}^{p} \mathbf{y}^{\mu} \cdot \left( \mathbf{x}^{\mu} \right)^{t} = \begin{pmatrix} y_1^1 & y_1^2 & y_1^3 & \cdots & y_n^p \\ y_2^1 & y_2^2 & y_2^3 & \cdots & y_2^p \\ \vdots & \vdots & \vdots & \cdots & \vdots \\ y_m^1 & y_m^2 & y_m^3 & \cdots & y_m^p \end{pmatrix}$$

Aprovechando la característica mostrada por el *Linear Associator* cuando los vectores de entrada $\mathbf{x}^{\mu}$ son ortonormales, y dado que, por la Definición 1, los vectores *one-hot* $\mathbf{h}^k$ con $k$ = 1, ..., $p$ son ortonormales, se puede obviar la fase de aprendizaje al evitar hacer las operaciones vectoriales realizadas por el *Linear Associator*, y simplemente colocando los vectores en orden para formar el *Linear Associator*.

Las etapas 2 y 4 corresponden a dos *Linear Associator* modificados, construidos con los vectores **y** y **x**, respectivamente, del conjunto fundamental.

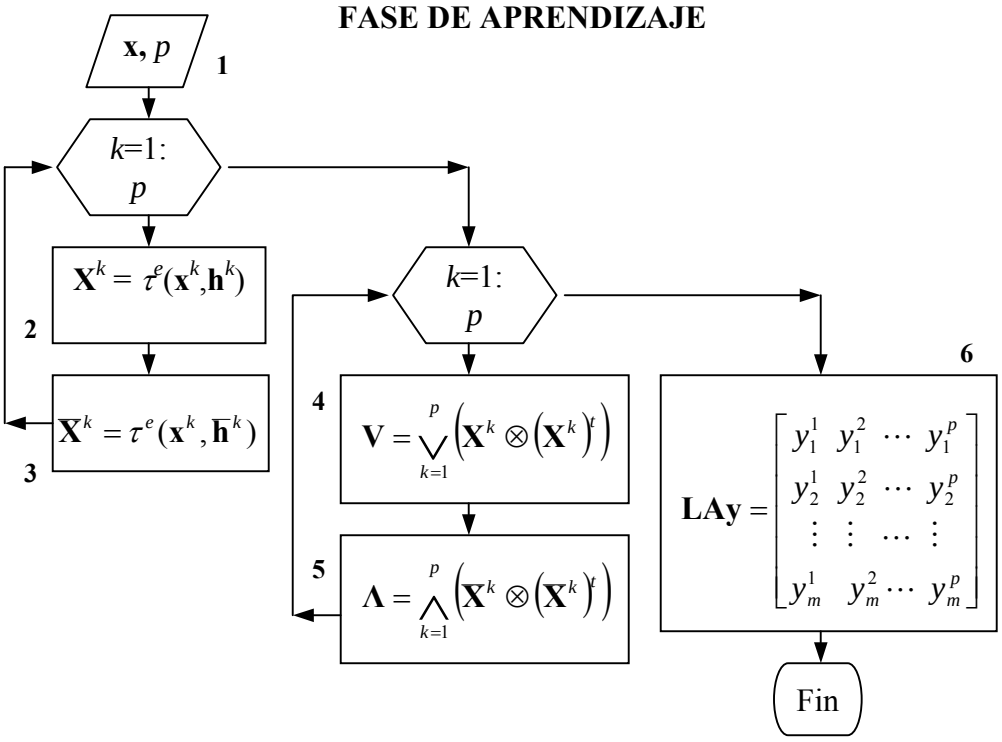
## 4.4 Algoritmo

En esta sección se describen paso a paso los procesos requeridos por la BAM Alfa-Beta tanto en la Fase de Aprendizaje, como en la Fase de Recuperación en el sentido de $\mathbf{x} \rightarrow \mathbf{y}$, algoritmo para las Etapas 1 y 2, y en el sentido $\mathbf{y} \rightarrow \mathbf{x}$, algoritmo para las Etapas 3 y 4.

Para finalizar este apartado, se presenta un ejemplo que ilustra la aplicación del algoritmo de las Etapas 1 y 2.

### 4.4.2 Algoritmo de las Etapas 1 y 2

El siguiente algoritmo describe los pasos requeridos por la memoria asociativa bidireccional Alfa-Beta para realizar la fase de aprendizaje y la fase de recuperación en el sentido de $\mathbf{x} \rightarrow \mathbf{y}$.



**Paso 1.** Como datos se tienen los *p* patrones de entrada **x**.

**Paso 2.** Se aplica la transformada de expansión vectorial del vector **x** para cada $\mathbf{X}^k$, utilizando como argumentos cada uno de los vectores de entrada $\mathbf{x}^k$ y cada vector *one-hot* $\mathbf{h}^k$.

**Paso 3.** Se aplica la transformada de expansión vectorial del vector **x** para cada $\overline{\mathbf{X}}^k$, utilizando como argumentos cada uno de los vectores de entrada $\mathbf{x}^k$ y cada vector *zero-hot* $\overline{\mathbf{h}}^k$.

**Paso 4.** Se crea una memoria asociativa Alfa-Beta *max* **V** con el conjunto fundamental

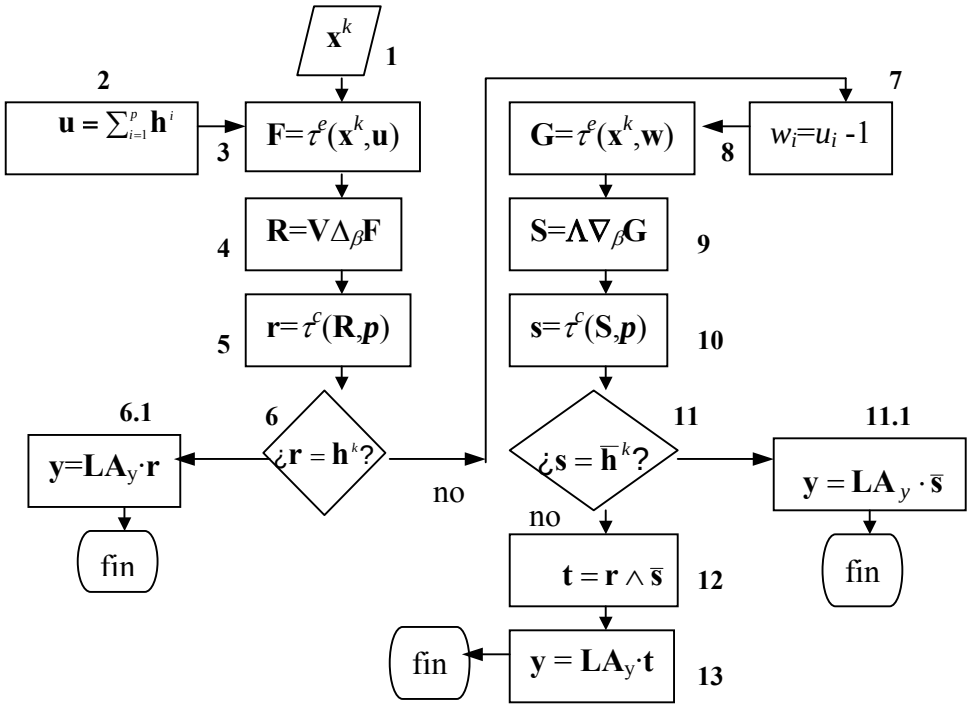
$$\left\{\left(\mathbf{X}^k, \mathbf{X}^k\right) \mid k = 1,..., p\right\}$$

**Paso 5.** Se crea una memoria asociativa Alfa-Beta *min* $\boldsymbol{\Lambda}$ con el conjunto fundamental

$$\left\{\left(\overline{\mathbf{X}}^k, \overline{\mathbf{X}}^k\right) \mid k = 1,..., p\right\}$$

**Paso 6.** Se crea una matriz **LAy**, que consiste de un *Linear Associator* modificado utilizando los patrones de salida **y**.

**FASE DE RECUPERACIÓN**




**Paso 1.** Presentar, a la entrada de la etapa 1, un vector del conjunto fundamental $\mathbf{x}^k \in A^n$ para algún índice $k \in \{1, ..., p\}$

**Paso 2.** Construir el vector $\mathbf{u} \in A^p$

**Paso 3.** Aplicar la transformada vectorial de expansión del vector **x** utilizando como argumentos el vector $\mathbf{x}^k$ y el vector **u**, para obtener **F**.

**Paso 4.** Operar la memoria autoasociativa Alfa-Beta *max* **V** con **F**, para obtener un vector **R.**

**Paso 5.** Aplicar la transformada de contracción del vector **R**, cuyos argumentos son el vector **R**, obtenido en el paso anterior, y *p* (número de patrones), para obtener el vector **r**.

**Paso 6. Si r** es un vector *one-hot*, entonces **r** es el *k-ésimo* vector *one-hot,* $\mathbf{h}^k$, (Basado en el Teorema 4.2) **entonces**.

**Paso 6.1** Se realiza la operación $\mathbf{LAy} \cdot \mathbf{r}$, lo que resultará en el $\mathbf{y}^k$ correspondiente. **Fin**.

**Si no,** entonces

**Paso 7.** Construir el vector $\mathbf{w} \in A^p$

**Paso 8.** Aplicar la transformada vectorial de expansión del vector **x** utilizando como argumentos el vector $\mathbf{x}^k$ y el vector **w**, para obtener **G**.

**Paso 9.** Operar la memoria autoasociativa Alfa-Beta *min* $\boldsymbol{\Lambda}$ con **G**, para obtener un vector **S.**

**Paso 10.** Aplicar la transformada de contracción del vector **S**, cuyos argumentos son el vector **S**, obtenido en el paso anterior, y *p* (número de patrones), para obtener el vector **s**.

**Paso 11. Si s** es un vector *zero-hot*, entonces **s** es el *k*-ésimo vector *zero-hot,* $\overline{\mathbf{h}}^k$, (Basado en el Teorema 4.4) **entonces**.

**Paso 11.1** Se realiza la operación $\mathbf{LAy} \cdot \overline{\mathbf{s}}$, lo que resultará en el $\mathbf{y}^k$ correspondiente. **Fin**. **Si no,** entonces

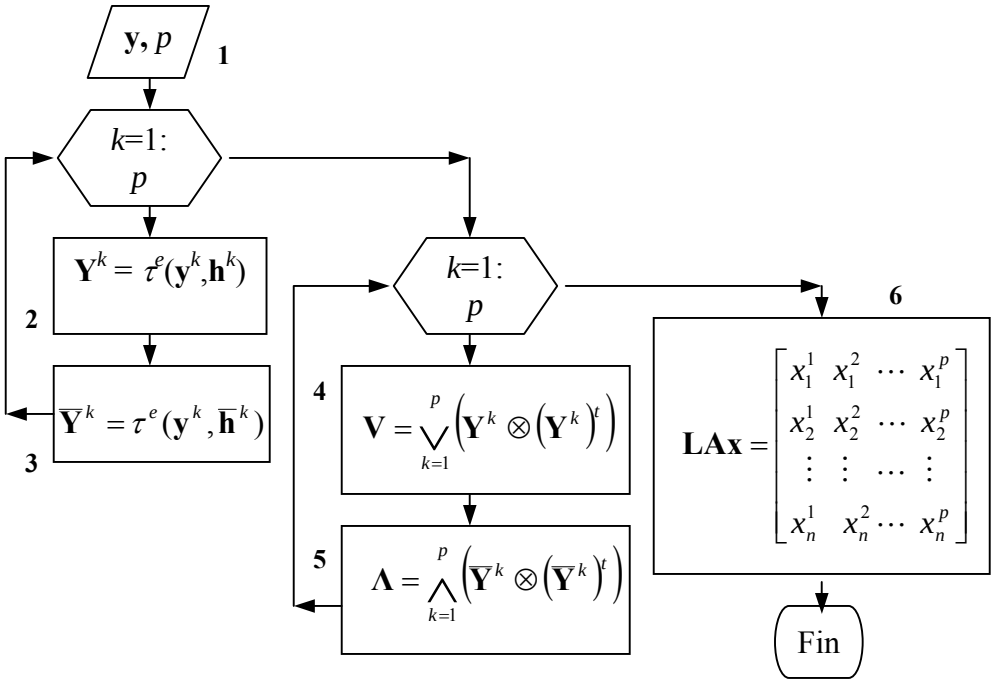
**Paso 12.** Realizar la operación AND lógica entre r y $\overline{\mathbf{s}}$ para obtener el vector **t**, el cual (Basado en el Teorema 4.5) será igual al *k*-ésimo vector *one-hot*.

**Paso 13.** Realizar la operación $\mathbf{LAy} \cdot \mathbf{t}$, para obtener el $\mathbf{y}^k$ correspondiente. **Fin**.

### 4.4.2 Algoritmo de las Etapas 3 y 4

El siguiente algoritmo describe los pasos requeridos por la memoria asociativa bidireccional Alfa-Beta para realizar la fase de aprendizaje y la fase de recuperación en el sentido de $\mathbf{y} \rightarrow \mathbf{x}$.

**FASE DE APRENDIZAJE**




**Paso 1.** Como datos se tienen los *p* patrones de salida **y**.

**Paso 2.** Se aplica la transformada de expansión vectorial del vector **Y** para cada $\mathbf{Y}^k$, utilizando como argumentos cada uno de los vectores de entrada $\mathbf{y}^k$ y cada vector *one-hot* $\mathbf{h}^k$.

**Paso 3.** Se aplica la transformada de expansión vectorial del vector $\overline{\mathbf{Y}}$ para cada $\overline{\mathbf{Y}}^k$, utilizando como argumentos cada uno de los vectores de entrada $\mathbf{y}^k$ y cada vector *zero-hot* $\overline{\mathbf{h}}^k$.

**Paso 4.** Se crea una memoria asociativa Alfa-Beta *max* **V** con el conjunto fundamental

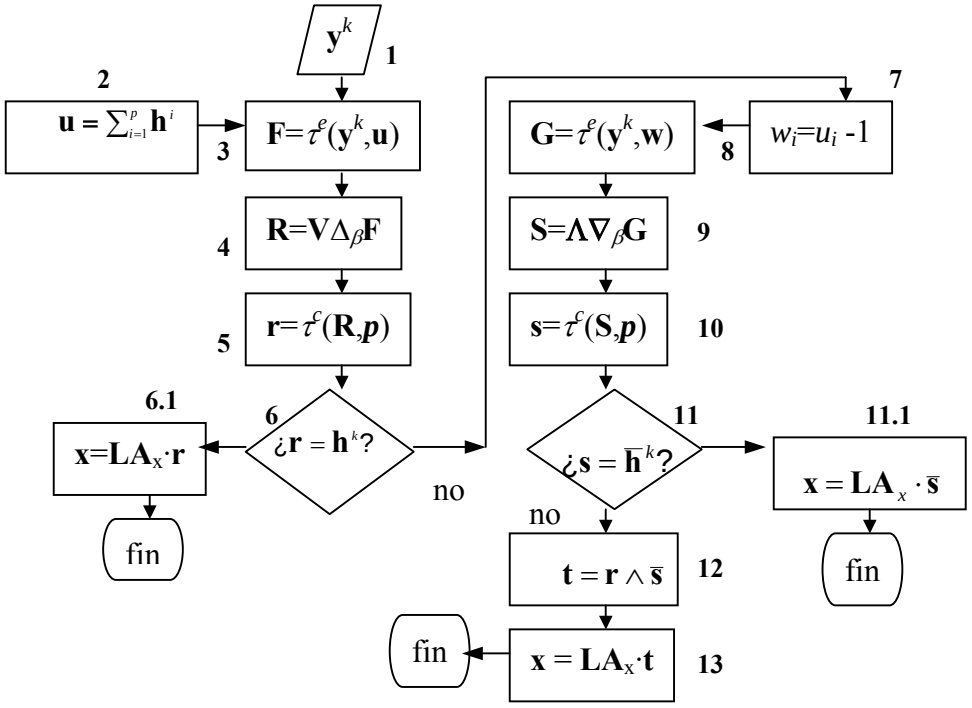
$$\left\{\left(\mathbf{Y}^k, \mathbf{Y}^k\right) \mid k = 1,..., p\right\}$$

**Paso 5.** Se crea una memoria asociativa Alfa-Beta *min* $\mathbf{\Lambda}$ con el conjunto fundamental

$$\left\{\left(\overline{\mathbf{Y}}^k, \overline{\mathbf{Y}}^k\right) \mid k = 1,..., p\right\}$$

**Paso 6.** Se crea una matriz **LAx**, que consiste de un *Linear Associator* modificado utilizando los patrones de entrada **x**.

## FASE DE RECUPERACIÓN




**Paso 1.** Presentar, a la entrada de la etapa 1, un vector del conjunto fundamental $\mathbf{y}^k \in A^m$ para algún índice $k \in \{1, ..., p\}$

**Paso 2.** Construir el vector $\mathbf{u} \in A^p$

**Paso 3.** Aplicar la transformada vectorial de expansión del vector **y** utilizando como argumentos el vector $\mathbf{y}^k$ y el vector **u**, para obtener **F**.

**Paso 4.** Operar la memoria autoasociativa Alfa-Beta *max* **V** con **F**, para obtener un vector **R.**

**Paso 5.** Aplicar la transformada de contracción del vector **R**, cuyos argumentos son el vector **R**, obtenido en el paso anterior, y *p* (número de patrones), para obtener el vector **r**.

**Paso 6. Si r** es un vector *one-hot*, entonces **r** es el *k-ésimo* vector *one-hot,* $\mathbf{h}^k$, (Basado en el Teorema 4.2) **entonces**.

**Paso 6.1** Se realiza la operación **LAx** · **r**, lo que resultará en el $\mathbf{x}^k$ correspondiente. **Fin**.

**Si no,** entonces

**Paso 7.** Construir el vector $\mathbf{w} \in A^p$

**Paso 8.** Aplicar la transformada vectorial de expansión del vector **y** utilizando como argumentos el vector $\mathbf{y}^k$ y el vector **w**, para obtener **G**.

**Paso 9.** Operar la memoria autoasociativa Alfa-Beta *min* $\boldsymbol{\Lambda}$ con **G**, para obtener un vector **S.**

**Paso 10.** Aplicar la transformada de contracción del vector **S**, cuyos argumentos son el vector **S**, obtenido en el paso anterior, y *p* (número de patrones), para obtener el vector **s**.

**Paso 11. Si s** es un vector *zero-hot*, entonces **s** es el *k*-ésimo vector *zero-hot,* $\overline{\mathbf{h}}^k$, (Basado en el Teorema 4.4) **entonces**.

**Paso 11.1** Se realiza la operación $\mathbf{LAx} \cdot \overline{\mathbf{s}}$, lo que resultará en el $\mathbf{x}^k$ correspondiente. **Fin**. **Si no,** entonces

**Paso 12.** Realizar la operación AND lógica entre r y $\bar{\mathbf{s}}$ para obtener el vector **t**, el cual (Basado en el Teorema 4.5) será igual al *k*-ésimo vector *one-hot*.

**Paso 13.** Realizar la operación $\mathbf{LAx} \cdot \mathbf{t}$, para obtener el $\mathbf{x}^k$ correspondiente. **Fin**.

### 4.4.3 Ejemplo ilustrativo de la Memoria Asociativa Bidireccional Alfa-Beta

Sean los siguientes 4 pares de patrones (*p*=4), los patrones de entrada, **x** con dimensión 4 (*n*=4) y los patrones de salida, **y** con dimensión 3 (*m*=3).

$$\mathbf{x}^1 = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}, \ \mathbf{y}^1 = \begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix}, \ \mathbf{x}^2 = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}, \ \mathbf{y}^2 = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}, \ \mathbf{x}^3 = \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \end{bmatrix}, \ \mathbf{y}^3 = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}, \ \mathbf{x}^4 = \begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \end{bmatrix}, \ \mathbf{y}^4 = \begin{bmatrix} 1 \\ 1 \\ 0 \end{bmatrix}$$

**FASE DE APRENDIZAJE**

- Se aplica la transformada vectorial de expansión del vector **x**, para obtener cada uno de los vectores $\mathbf{X}^k$ y $\overline{\mathbf{X}}^k$

$$\mathbf{X} = \tau^e(\mathbf{x}, \mathbf{h}) \qquad \overline{\mathbf{X}} = \tau^e(\mathbf{x}, \overline{\mathbf{h}})$$

$$\mathbf{X}^1 = \begin{bmatrix}1\\1\\1\\1\\1\\0\\0\\0\end{bmatrix} \quad \mathbf{X}^2 = \begin{bmatrix}0\\0\\0\\0\\0\\1\\0\\0\end{bmatrix} \quad \mathbf{X}^3 = \begin{bmatrix}0\\0\\1\\0\\0\\0\\1\\0\end{bmatrix} \quad \mathbf{X}^4 = \begin{bmatrix}1\\1\\0\\0\\0\\0\\0\\1\end{bmatrix} \qquad \overline{\mathbf{X}}^1 = \begin{bmatrix}1\\1\\1\\1\\0\\1\\1\\1\end{bmatrix} \quad \overline{\mathbf{X}}^2 = \begin{bmatrix}0\\0\\0\\0\\1\\0\\1\\1\end{bmatrix} \quad \overline{\mathbf{X}}^3 = \begin{bmatrix}0\\0\\1\\0\\1\\1\\0\\1\end{bmatrix} \quad \overline{\mathbf{X}}^4 = \begin{bmatrix}1\\1\\0\\0\\1\\1\\1\\0\end{bmatrix}$$

**Vectores *one-hot*** **Vectores *zero-hot***

- Se generan las memorias autoasociativas Alfa-Beta *max* $\mathbf{V} = \bigvee_{k=1}^{4} \left( \mathbf{X}^k \otimes \left( \mathbf{X}^k \right)^t \right)$ y *min* $\mathbf{\Lambda} = \bigwedge_{k=1}^{4} \left( \overline{\mathbf{X}}^k \otimes \left( \overline{\mathbf{X}}^k \right)^t \right)$

$$\mathbf{V} = \begin{bmatrix}1&1&2&2&2&2&2&1\\1&1&2&2&2&2&2&2\\2&2&1&2&2&2&2&2\\1&1&1&1&1&2&2&2\\1&1&1&1&1&2&2&2\\2&2&2&2&2&1&2&2\\2&2&1&2&2&2&1&2\\1&1&2&2&2&2&2&1\end{bmatrix} \text{ y } \mathbf{\Lambda} = \begin{bmatrix}1&1&0&1&0&0&0&0\\1&1&0&1&0&0&0&0\\0&0&1&1&0&0&0&0\\0&0&0&1&0&0&0&0\\0&0&0&0&1&0&0&0\\1&1&1&1&0&1&0&0\\1&1&0&1&0&0&1&0\\0&0&1&1&0&0&0&1\end{bmatrix}$$

Lema 4.3 Lema 4.5

- Se crea el *Linear Associator* Modificado utilizando los patrones de salida **y**

$$\mathbf{LAy} = \begin{bmatrix}1&1&0&1\\0&0&1&1\\1&0&0&0\end{bmatrix}$$

**FASE DE RECUPERACIÓN**

- Se presenta a la entrada de la BAM Alfa-Beta el patrón $\mathbf{x}^3$, se construye el vector **u** y se construye la expansión

$$\mathbf{x}^3 = \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \end{bmatrix} \qquad \mathbf{u} = \sum_{i=1}^{4} \mathbf{h}^i = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} \qquad \boxed{\mathbf{F} = \tau^e(\mathbf{x}^3, \mathbf{u})} \qquad \mathbf{F} = \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}$$

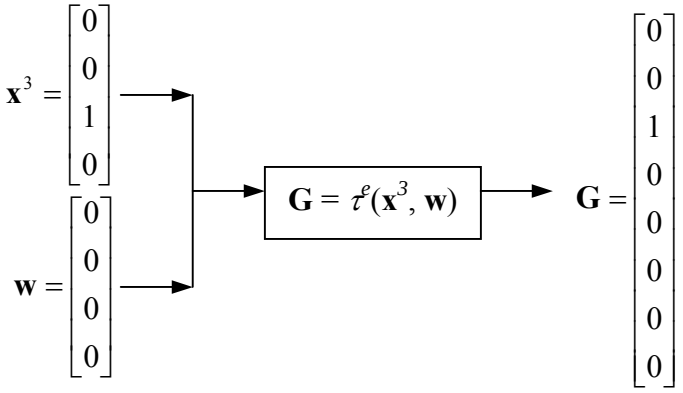
- Se opera la memoria autoasociativa Alfa-Beta *max* **V** con **F**

$$\mathbf{R} = \mathbf{V} \Delta_\beta \mathbf{F} = \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 1 \\ 1 \\ 0 \end{bmatrix}$$ Lema 4.1

- Se realiza la contracción: $\mathbf{r} = \tau^c(\mathbf{R}, 4) = \begin{bmatrix} 0 \\ 1 \\ 1 \\ 0 \end{bmatrix}$ Teorema 4.2

Debido a que **r** no es un vector *one-hot*, entonces se continua con el algoritmo

- Se construye el vector **u** y con el patrón $\mathbf{x}^3$, se construye la expansión




- Se opera la memoria autoasociativa Alfa-Beta *min* $\boldsymbol{\Lambda}$ con $\mathbf{G}$

$$\mathbf{S} = \boldsymbol{\Lambda}\nabla_{\beta}\mathbf{G} = \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 1 \\ 0 \\ 1 \end{bmatrix}$$ Lema 4.2

- Se realiza la contracción $\mathbf{s} = \tau^{c}(\mathbf{S},4) = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 1 \end{bmatrix}$ Teorema 4.4

El vector **s** no es un vector *zero-hot*, por lo que se continua con el proceso.

- Se realiza la operación AND entre el vector **r** y el vector negado de **s**, $\bar{s}$ , para obtener el vector **t**

$$\mathbf{t} = \begin{bmatrix} 0 \\ 1 \\ 1 \\ 0 \end{bmatrix} \Lambda \begin{bmatrix} 1 \\ 0 \\ 1 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \end{bmatrix}$$ **Teorema 4.5**

Con esta operación se obtiene el tercer vector *one-hot*

- Se obtiene el vector $\mathbf{y}^3$ mediante

$$\mathbf{y}_{rec} = \mathbf{LAy} \cdot \mathbf{t} = \begin{bmatrix} 1 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} = \mathbf{y}^3$$

## 4.5 Complejidad del Algoritmo de la BAM Alfa-Beta

Un algoritmo es un conjunto finito de instrucciones precisas para la realización de un cálculo o para resolver un problema [66]. En general, se acepta que un algoritmo provee una solución satisfactoria cuando produce una respuesta correcta y cuando es eficiente.

Una medida de la eficiencia es el tiempo usado por la computadora para resolver un problema utilizando un algoritmo dado. Una segunda medida de eficiencia es la cantidad de memoria requerida para implementar el algoritmo cuando los valores de entrada son de un tamaño específico.

El análisis del tiempo requerido para resolver un problema de un tamaño en particular implica la **complejidad en tiempo** del algoritmo. El análisis de la memoria requerida por la computadora implica la **complejidad en espacio** del algoritmo.

En las siguientes secciones se describe la complejidad del algoritmo que presenta la BAM Alfa-Beta. En la primera parte, se analiza la complejidad en espacio tanto para la BAM de Kosko como para la Alfa-Beta, para efectos de comparación. En la segunda sección, se analiza la complejidad en tiempo sólo para el algoritmo del modelo propuesto en este trabajo de tesis, dado que no es posible realizar la correspondiente comparación con la BAM de Kosko, en virtud de que ésta funciona con base en un algoritmo iterativo, lo que impide conocer de antemano el número de iteraciones.

### 4.5.1 Complejidad en espacio

**BAM de Kosko**

Los $p$ patrones de entrada $\mathbf{x}$, con dimensión $n$, se guardan en una matriz cuya dimensión será $p$ x $n$. Dado que $\mathbf{x} \in \{-1, 1\}$, entonces estos se representan con variables enteras que ocupan 2 bytes. El número total de bytes será:

Bytes_x = $2pn$

Los $p$ patrones de entrada $\mathbf{y}$, con dimensión $m$, se guardan en una matriz cuya dimensión será $p$ x $m$. Dado que $\mathbf{y} \in \{-1, 1\}$, entonces estos se representan con variables enteras que ocupan 2 bytes. El número total de bytes será:

Bytes_x = $2pm$

La matriz de correlación **M** tiene dimensiones de $m$ x $n$ y dado que $\mathbf{M} \in Z$, entonces debe representarse con variables enteras que ocupan 2 bytes. El número total de bytes será:

Bytes_M = $2mn$

La matriz de correlación transpuesta $\mathbf{M}^t$ tiene dimensiones de $n$ x $m$ y dado que $\mathbf{M}^t \in Z$, entonces debe representarse con variables enteras que ocupan 2 bytes. El número total de bytes será:

Bytes_Mt = $2nm$

Se necesitan dos vectores que guarden el estado anterior en la fase de recuperación, uno para **x** y otro para **y**. Dadas las dimensiones de estos vectores, el número de bytes será:

Bytes_xy = $2n + 2m = 2(n+m)$

Se necesitan otros dos vectores para guardar los patrones recuperados, tanto para **x** como para **y**. Dadas las dimensiones de estos vectores, el número de bytes será:

Bytes_xyr = $2n + 2m = 2(n+m)$

El total de bytes requeridos para implementar la BAM de Kosko es de:

Total = Bytes_x + Bytes_x + Bytes_M + Bytes_Mt + Bytes_xy + Bytes_xyr

Total = $2pn + 2pm + 2mn + 2nm + 2(n+m) + 2(n+m) = 2p(n+m) + 4mn + 4(n+m)$

Total = $(n+m)(2p + 4) + 4mn$

**BAM Alfa-Beta**

Se necesitan una matriz para guardar los $p$ patrones **x**. La matriz tendrá dimensiones de $p$ x $(n+p)$. En la misma matriz se guardan los patrones de entrada y los vectores añadidos tanto del *one-hot* como del *zero-hot*. Dado que $\mathbf{x} \in \{0, 1\}$, entonces estos valores se pueden representar con variables de tipo caracter que ocupan 1 byte. El número total de bytes será:

Bytes_x = $p(n+ p)$

Se necesita una matriz para guardar los $p$ patrones **y**. La matriz tendrá dimensiones de $p$ x $(m+p)$. En la misma matriz se guardan los patrones de salida y los vectores añadidos tanto del *one-hot* como del *zero-hot*. Dado que $\mathbf{y} \in \{0, 1\}$, entonces estos valores se pueden representar con variables de tipo caracter que ocupan 1 byte. El número total de bytes será:

Bytes_y = $p(m+ p)$

En la fase de aprendizaje se necesitan 4 matrices. Dos para las memoria autoasociativas Alfa-Beta tipo max , Vx y Vy, y dos para las memorias autoasociativas Alfa-Beta tipo min, Λx y Λy. Vx y Λx tienen dimensiones ($n$+$p$) x ($n$+$p$) y Vy y Λy tiene dimensiones ($m$+$p$) x ($m$+$p$). Dado que estas matrices contienen sólo números enteros positivos, entonces los valores de sus componentes se pueden representar con variables de tipo carácter que ocupan 1 byte. El número total de bytes será:

$$\text{Bytes\_VxΛx} = 2(n+p)^2$$

$$\text{Bytes\_VyΛy} = 2(m+p)^2$$

Se utiliza el vector en donde se guarda el vector *one-hot* recuperado, cuya dimensión es de $p$. Dado que las componentes de cualquier vector one-hot toman los valores de 0 y 1, entonces estos valores se pueden representar con variables de tipo carácter que ocupan 1 byte. El número total de bytes será

$$\text{Bytes\_vr} = p$$

El total de bytes requeridos para implementar una BAM Alfa-Beta es:

$$\text{Total} = \text{Bytes\_x} + \text{Bytes\_y} + \text{Bytes\_VxΛx} + \text{Bytes\_VyΛy} + \text{Bytes\_vr}$$

$$\text{Total} = p(n+p) + p(m+p) + 2(n+p)^2 + 2(m+p)^2 + p$$

$$\text{Total} = p[(n+p) + (m+p)] + 2[(n+p)^2 + (m+p)^2] + p$$

$$\text{Total} = p(n + m + 2p) + 2[(n+p)^2 + (m+p)^2] + p$$

### 4.5.2 Complejidad en tiempo

La complejidad en tiempo de un algoritmo se puede expresar en términos del número de operaciones usadas por el algoritmo cuando la entrada tiene un tamaño en particular. Las operaciones utilizadas para medir la complejidad en tiempo pueden ser la comparación de enteros, la suma de enteros, la división de enteros, asignaciones de variables, comparaciones lógicas, o cualquier otra operación elemental.

Se define lo siguiente:

OE: operación elemental
*n_pares*: es el número de pares de patrones asociados

$n$: es la dimensión de los patrones más la adición de los vectores *one-hot* o *zero-hot*
Se analiza el algoritmo de la fase de aprendizaje porque es la parte de todo el algoritmo que requiere de más operaciones elementales.

Fase de Recuperación

```
u = 0; (1)
while(u<n_pares) (2)
 {
        i = 0;  (3)
         while(i<n)  (4)
         {
                j = 0;  (5)
                 while(j<n)  (6)
                 {
                        if(y[u][i]==0 && y[u][j]==0) (7)
                                t=1;  (8)
                        else if(y[u][i]==0 && y[u][j]==1) (9a)
                                t=0;
                        else if(y[u][i]==1 && y[u][j]==0) (9b)
                                t=2;
                        else            t=1;
                        if(u==0) (10)
                                Vy[i][j]=t; (11)
                        else
                                if(Vy[i][j]<t)  (12)
                                        Vy[i][j]=t;  (13)
                        j++;  (14)
                 }
                 i++;  (15)
         }
        u++;   (16)
 }
```

(1) 1 OE, asignación
(2) n_pares OE, comparaciones
(3) n_pares OE, asignación
(4) n_pares**n* OE, comparaciones
(5) n_pares**n* OE, asignación
(6) n_pares**n***n* OE, comparaciones

(7a) n_pares**n***n* OE, comparaciones: y[u][i]==0
(7b) n_pares**n***n* OE, operador relacional AND: &&
(7c) n_pares**n***n* OE, comparaciones: y[u][j]==0
(8) Siempre se va a realizar una asignación a la variable t, n_pares*n*n OE
(9) Ambas sentencias if (a y b) tiene la mitad de probabilidad de ejecutarse, n_pares**n**(*n*/2)
(10) n_pares**n***n* OE, comparaciones
(11) Esta asignación sólo se realiza una vez, 1 OE
(12) (n_pares**n***n*)-1 OE, comparaciones
(13) tiene la mitad de probabilidad de ejecutarse la asignación, n_pares**n**(*n*/2)
(14) n_pares**n***n* OE, incremento

(15) n_pares*$n$ OE, incremento
(16) n_pares OE, incremento

El número total de OE es:

Total = 1+n_pares+n_pares+n_pares*$n$+n_pares*$n$+n_pares*$n$*$n$+n_pares*$n$*$n$+ n_pares*$n$*$n$+ n_pares*$n$*$n$+ n_pares*$n$*$n$+ n_pares*$n$*($n$/2)+ n_pares*$n$*$n$+1+[(n_pares*$n$*$n$)-1]+ n_pares*$n$*($n$/2)+ n_pares*$n$*$n$+ n_pares*$n$+ n_pares

Total = 2+3n_pares+3 n_pares*$n$+7 n_pares*$n$*$n$+2[n_pares*$n$*($n$/2)]+ [(n_pares*$n$*$n$)-1]

Total = 2+n_pares(3+3$n$+8$n^2$)+ n_pares*$n^2$-1

Total = 1+n_pares(3+3$n$+9 $n^2$)

Del total de OE obtenida, se fija n_pares en el valor de 50, lo que resulta en una función que sólo depende del tamaño de los patrones:

$$f(n) = 1+50(3+3n+9n^2)$$

Para analizar la factibilidad del algoritmo es necesario entender que tan rápido crece la función obtenida conforme aumenta el valor de $n$. Para este cometido, se utilizará la notación BIG-O [66], que se define a continuación.

Sea $f$ y g funciones de un conjunto de enteros o de un conjunto de números reales a un conjunto de números reales. Se dice que $f(x)$ es $O(g(x))$ si existen dos constantes $C$ y $k$ tal que:

$$|f(x)| \leq C\,|g(x)| \quad \text{cuando } x > k$$

Ahora, el número de operaciones elementales obtenidas de nuestro algoritmo fue:

$$f(n) = 1+50(3+3n+9\ n^2)$$

se debe encontrar una función $g(x)$, $C$ y $k$, que cumplan con la desigualdad.

Se propone

$$50(3n^2+3n^2+9n^2) = 150n^2+150n^2+450n^2 = 750n^2$$

Entonces si $g(n) = n^2$, $C = 750$ y $k = 1$, tenemos que

$$|f(n)| \leq 750\,|g(n)| \quad \text{cuando } n > 1, \text{ por lo tanto } O(n^2).$$

La figura 4.6 se muestra las gráficas de $f(n)$ y $O(n^2)$. Se puede observar que la función $g(n)$ propuesta cumple con la desigualdad, con lo que se concluye que el algoritmo de la BAM Alfa-Beta es $O(n^2)$.

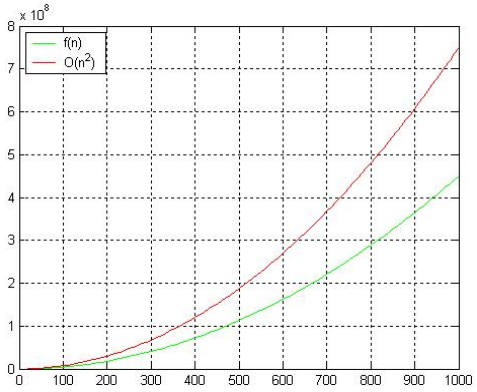



**Figura 4.6** Gráfica de $f(n)$ y $O(n^2)$. Se comprueba la desigualdad, por lo tanto, el algoritmo es $O(n^2)$.

# CAPÍTULO 5

# Resultados

En este capítulo se presenta una comparación de la BAM Alfa-Beta con la BAM de Kosko y la BAM de Wang [28] mediante ejemplos numéricos. Se presentan, además, los resultados experimentales que se realizaron al tomar conjuntos fundamentales formados por imágenes, para la comparación entre la BAM Alfa-Beta y el método de Shen y Cruz [18] que utiliza un algoritmo genético.

Adicionalmente se muestra la superioridad del nuevo modelo, motivo de esta tesis, al desarrollar tres aplicaciones que incluyen aspectos de interés en algunas áreas de la actividad humana, como lo es el reconocimiento automático de huellas digitales. De este modo se evidencia que el fundamento teórico de las Memorias Asociativas Bidireccionales Alfa-Beta puede redundar en aplicaciones prácticas de interés para los investigadores internacionales.

La primera aplicación, presentada en la sección 5.2, consta de 4 conjuntos de figuras, en donde cada conjunto tiene 26 figuras monocromáticas. La aplicación permite elegir las figuras que se deseen para crear un conjunto fundamental que servirá de base en la operación de las Memorias Asociativas Bidireccionales Alfa-Beta.

En la segunda aplicación, presentada en la sección 5.3, se realizan experimentos con 40 huellas digitales que se asocian a 40 números.

La tercera aplicación, que se presenta en la sección 5.4, es un traductor inglés-español/español-inglés. Esta aplicación asocia 120 palabras en inglés con 120 palabras en español. Además, el traductor puede recibir palabras con errores ortográficos o incompletas y, en algunos casos, recupera la traducción correcta.

La implementación del algoritmo se realizó con el compilador Visual C++ 6.0.

En la sección 5.5 se presentan el comportamiento del nuevo modelo de BAM Alfa-Beta ante patrones afectados por los tres tipos de ruido: aditivo, sustractivo y mezclado.

## 5.1 Comparación de Memorias Asociativas Bidireccionales

En el artículo de Wang *et al* [28] se presenta un ejemplo ilustrativo con tres parejas de patrones, es decir, $p = 3$. Las parejas de vectores son:

$A_1$ = (100111000), $B_1$ = (111000010)

$A_2$ = (011100111), $B_2$ = (100000001)

$A_3$ = (101011011), $B_3$ = (010100101)

Dado que las BAM, tanto de Kosko como de Wang, trabajan con patrones bipolares, entonces:

$X_1$ = (1 –1 –1 1 1 1 –1 –1 –1), $Y_1$ = (1 1 1 –1 –1 –1 –1 1 –1)

$X_2$ = (–1 1 1 1 –1 –1 1 1 1), $Y_2$ = (1 –1 –1 –1 –1 –1 –1 –1 1)

$X_3$ = (1 –1 1 –1 1 1 –1 1 1), $Y_3$ = (–1 1 –1 1 –1 –1 1 –1 1)

La matriz $M$ de Kosko se calcula para $p$ = 3:

$$M = X_1^T Y_1 + X_2^T Y_2 + X_3^T Y_3$$

$$M = \begin{bmatrix} -1 & 3 & 1 & 1 & -1 & -1 & 1 & 1 & -1 \\ 1 & -3 & -1 & -1 & 1 & 1 & -1 & -1 & 1 \\ -1 & -1 & -3 & 1 & -1 & -1 & 1 & -1 & 3 \\ 3 & -1 & 1 & -3 & -1 & -1 & -3 & 1 & -1 \\ -1 & 3 & 1 & 1 & -1 & -1 & 1 & 1 & -1 \\ -1 & 3 & 1 & 1 & -1 & -1 & 1 & 1 & -1 \\ 1 & -3 & -1 & -1 & 1 & 1 & -1 & -1 & 1 \\ -1 & -1 & -3 & 1 & -1 & -1 & 1 & -3 & 3 \\ -1 & -1 & -3 & 1 & -1 & -1 & 1 & -3 & 3 \end{bmatrix}$$

Supongamos que se comienza con $\alpha = X_2$. Entonces $\alpha M = X_2 M$ y tiene el valor de:

$\alpha M$ = (5 –19 –13 –5 1 1 –5 –13 13)

Entonces $\phi(\alpha M) = \beta'$ = (1 –1 –1 –1 1 1 –1 –1 1) y $\alpha' = \phi(\beta' M^T)$

$\beta' M^T$ = (–11 11 5 5 –11 –11 11 5 5)

Entonces $\alpha' = \phi(\beta' M^T)$ = (–1 1 1 1 –1 –1 1 1 1) = $X_2 = \alpha$.

El uso de $\alpha'$ producirá $\beta'' = \beta'$. Por lo tanto, el ciclo ha terminado con $\alpha_F = \alpha = X_2$ = (–1 1 1 1 –1 –1 1 1 1) y $\beta_F = \beta'$ = (1 –1 –1 –1 1 1 –1 –1 1). Su puede observar, que $\beta_F \neq Y_2$ y por tanto, la pareja ($X_2$, $Y_2$) no se recupera correctamente mediante proceso de decodificación de Kosko.

Wang utiliza su método del entrenamiento múltiple para recuperar las tres parejas correctamente. Debido a que la BAM de Kosko no recupera la pareja ($X_2$, $Y_2$) entonces propone el factor de entrenamiento de 2 para esa pareja, entonces el nuevo valor de $M$ es:

$$M = X_1^T Y_1 + 2X_2^T Y_2 + X_3^T Y_3$$

Con esta nueva matriz $M$ se recupera correctamente la pareja ($X_2$, $Y_2$); sin embargo, esto genera que la pareja ($X_1$, $Y_1$) no se recupere de manera correcta. Este problema se resuelve aplicando el factor de entrenamiento de 2 a la pareja ($X_1$, $Y_1$). La nueva matriz $M$ es:

$$M = 2X_1^T Y_1 + 2X_2^T Y_2 + X_3^T Y_3$$

Con esta matriz se recuperan de manera perfecta las tres parejas de patrones. Cabe hacer notar que este método garantiza la recuperación del patrón al que se le aplica el entrenamiento múltiple. Este método muestra la clara desventaja de que la aplicación del entrenamiento múltiple a una de las parejas produce errores en la recuperación de las otras parejas; además, no existe un algoritmo que proponga de manera directa el valor del factor de entrenamiento. La aplicación del entrenamiento múltiple, a cada término para el cálculo de la matriz $M$, se realiza mediante ensayo y error.

Ahora utilizaremos el algoritmo de la BAM Alfa-Beta para el mismo ejemplo. Recordemos que nuestro método trabaja con patrones binarios.

Para cada índice $k \in \{1, ..., p\}$ se realiza la expansión $\mathbf{X}^k = \tau^e(\mathbf{A}_k, \mathbf{h}^k)$ y $\mathbf{Y}^k = \tau^e(\mathbf{B}_k, \mathbf{h}^k)$, donde $\mathbf{h}^k$ es el $k$-ésimo vector *one-hot*,

$$\mathbf{X}^1 = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 1 \\ 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \end{bmatrix}, \quad \mathbf{X}^2 = \begin{bmatrix} 0 \\ 1 \\ 1 \\ 1 \\ 0 \\ 0 \\ 1 \\ 1 \\ 1 \\ 0 \\ 1 \\ 0 \end{bmatrix}, \quad \mathbf{X}^3 = \begin{bmatrix} 1 \\ 0 \\ 1 \\ 0 \\ 1 \\ 1 \\ 0 \\ 1 \\ 1 \\ 0 \\ 0 \\ 1 \end{bmatrix} \quad \text{y} \quad \mathbf{Y}^1 = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 1 \\ 0 \\ 0 \end{bmatrix}, \quad \mathbf{Y}^2 = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 1 \\ 0 \end{bmatrix}, \quad \mathbf{Y}^3 = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 1 \\ 0 \\ 0 \\ 1 \\ 0 \\ 1 \\ 0 \\ 0 \\ 1 \end{bmatrix}$$

Con los vectores $\mathbf{X}^k$ con $k$ = 1, 2, 3 se forma la memoria autoasociativa Alfa-Beta *max* **V**x. De manera similar, pero ahora con los vectores $\mathbf{Y}^k$ con $k$ = 1, 2, 3 se forma la memoria autoasociativa Alfa-Beta *max* **V**y.

Para cada índice $k \in \{1, ..., p\}$ se realiza la expansión $\overline{\mathbf{X}}^k = \tau^e(\mathbf{A}_k, \overline{\mathbf{h}}^k)$ y $\overline{\mathbf{Y}}^k = \tau^e(\mathbf{B}_k, \overline{\mathbf{h}}^k)$, donde $\overline{\mathbf{h}}^k$ es el $k$-ésimo vector *zero-hot*.

$$\overline{\mathbf{X}}^1 = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 1 \\ 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \end{bmatrix}, \quad \overline{\mathbf{X}}^2 = \begin{bmatrix} 0 \\ 1 \\ 1 \\ 1 \\ 0 \\ 0 \\ 1 \\ 1 \\ 1 \\ 1 \\ 0 \\ 1 \end{bmatrix}, \quad \overline{\mathbf{X}}^3 = \begin{bmatrix} 1 \\ 0 \\ 1 \\ 0 \\ 1 \\ 1 \\ 0 \\ 1 \\ 1 \\ 1 \\ 1 \\ 0 \end{bmatrix} \quad \text{y} \quad \overline{\mathbf{Y}}^1 = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 1 \\ 1 \end{bmatrix}, \quad \overline{\mathbf{Y}}^2 = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \\ 0 \\ 1 \end{bmatrix}, \quad \overline{\mathbf{Y}}^3 = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 1 \\ 0 \\ 0 \\ 1 \\ 0 \\ 1 \\ 1 \\ 1 \\ 0 \end{bmatrix}$$

Con los vectores $\overline{\mathbf{X}}^k$ con $k$ = 1, 2, 3 se forma la memoria autoasociativa Alfa-Beta *min* **Λ**x. De manera similar, pero ahora con los vectores $\mathbf{Y}^k$ con $k$ = 1, 2, 3 se forma la memoria autoasociativa Alfa-Beta *min* **Λ**y.

Se crean dos *Linear Associator* modificados con los patrones $A_k$ y $B_k$ con $k$ = 1, 2, 3 de la siguiente forma: se toma cada uno de los vectores columna $A_k$ y se crea la matriz **LAx**, también se toma cada uno de los vectores columna y se crea la matriz **LAy**:

$$\mathbf{LAx} = \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 0 \\ 0 & 1 & 1 \\ 1 & 1 & 0 \\ 1 & 0 & 1 \\ 1 & 0 & 1 \\ 0 & 1 & 0 \\ 0 & 1 & 1 \\ 0 & 1 & 1 \end{bmatrix} \quad \text{y} \quad \mathbf{LAy} = \begin{bmatrix} 1 & 1 & 0 \\ 1 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 1 \end{bmatrix}$$

Al presentarle el patrón $A_1$ a la BAM Alfa-Beta, el proceso de recuperación es el siguiente:

Se crea el vector **F** que es una versión ruidosa, con ruido aditivo, del patrón $\mathbf{X}_1$ ,

$$\mathbf{F} = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 1 \\ 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \\ 1 \end{bmatrix} \quad \text{y se le presenta a la matriz } \mathbf{V}x, \quad \mathbf{R} = \mathbf{V}x\Lambda_{\beta}\mathbf{F} = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 1 \\ 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \end{bmatrix}$$

Realizamos la contracción $\mathbf{r} = \tau^{c}(\mathbf{R},n)$, entonces:

$$\mathbf{r} = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}, \text{ el cual corresponde al 1er. vector } \textit{one-hot}$$

El vector **r** se le presenta a la matriz **LAy** y se realiza la siguiente operación:

$$\mathbf{LAy} \cdot \mathbf{r} = \begin{bmatrix} 1 & 1 & 0 \\ 1 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \end{bmatrix} = \mathbf{B}_1$$

Se observa que el patrón $B_1$ se recupera correctamente.

Ahora le presentamos el patrón $A_2$ a la BAM Alfa-Beta, entonces el proceso de recuperación es el siguiente:

Se crea el vector **F** que es una versión ruidosa, con ruido aditivo, del patrón $\mathbf{X}_2$ ,

$$\mathbf{F} = \begin{bmatrix} 0 \\ 1 \\ 1 \\ 1 \\ 0 \\ 0 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} \quad \text{y se le presenta a la matriz } \mathbf{V}\text{x}, \quad \mathbf{R} = \mathbf{V}x\Lambda_{\beta}\mathbf{F} = \begin{bmatrix} 0 \\ 1 \\ 1 \\ 1 \\ 0 \\ 0 \\ 1 \\ 1 \\ 1 \\ 0 \\ 1 \\ 0 \end{bmatrix}$$

Realizamos la contracción $\mathbf{r} = \tau^{c}(\mathbf{R},n)$, entonces:

$$\mathbf{r} = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}, \text{ el cual corresponde al 2º vector } \textit{one-hot}$$

El vector **r** se le presenta a la matriz **LAy** y se realiza la siguiente operación:

$$\mathbf{LAy} \cdot \mathbf{r} = \begin{bmatrix} 1 & 1 & 0 \\ 1 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 1 \end{bmatrix} \cdot \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} = \mathbf{B}_2$$

Se observa que el patrón $B_2$ se recupera correctamente.

Ahora le presentamos el patrón $A_3$ a la BAM Alfa-Beta, entonces el proceso de recuperación es el siguiente:

Se crea el vector **F** que es una versión ruidosa, con ruido aditivo, del patrón $\mathbf{X}_3$ ,

$$\mathbf{F} = \begin{bmatrix} 1 \\ 0 \\ 1 \\ 0 \\ 1 \\ 1 \\ 0 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} \quad \text{y se le presenta a la matriz } \mathbf{V}x, \quad \mathbf{R} = \mathbf{V}x\Lambda_{\beta}\mathbf{F} = \begin{bmatrix} 1 \\ 0 \\ 1 \\ 0 \\ 1 \\ 1 \\ 0 \\ 1 \\ 1 \\ 0 \\ 0 \\ 1 \end{bmatrix}$$

Realizamos la contracción $\mathbf{r} = \tau^{c}(\mathbf{R},n)$, entonces:

$$\mathbf{r} = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}, \text{ el cual corresponde al 3er. vector } \textit{one-hot}$$

El vector **r** se le presenta a la matriz **LAy** y se realiza la siguiente operación:

$$\mathbf{LAy} \cdot \mathbf{r} = \begin{bmatrix} 1 & 1 & 0 \\ 1 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 1 \end{bmatrix} \cdot \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 1 \\ 0 \\ 0 \\ 1 \\ 0 \\ 1 \end{bmatrix} = \mathbf{B}_3$$

Se observa que el patrón $B_3$ se recupera correctamente.

Haciendo uso de la propiedad de bidireccionalidad de la BAM, a continuación le presentaremos, como entrada, los patrones $B_k$ para obtener sus $A_k$ correspondientes, con $k$ = 1, 2, 3.

Se le presenta el patrón $B_1$ a la BAM Alfa-Beta, el proceso de recuperación es el siguiente:
Se crea el vector **F** que es una versión ruidosa, con ruido aditivo, del patrón $\mathbf{Y}_1$ ,

$$\mathbf{F} = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 1 \\ 1 \\ 1 \end{bmatrix} \quad \text{y se le presenta a la matriz } \mathbf{V}\text{x}, \quad \mathbf{R} = \mathbf{V}x\Lambda_{\beta}\mathbf{F} = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 1 \\ 0 \\ 0 \end{bmatrix}$$

Realizamos la contracción $\mathbf{r} = \tau^c(\mathbf{R},n)$, entonces:

$$\mathbf{r} = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}, \text{ el cual corresponde al 1er. vector } \textit{one-hot}$$

El vector **r** se le presenta a la matriz **LAx** y se realiza la siguiente operación:

$$\mathbf{LAx} \cdot \mathbf{r} = \begin{bmatrix} 1\ 0\ 1 \\ 0\ 1\ 0 \\ 0\ 1\ 1 \\ 1\ 1\ 0 \\ 1\ 0\ 1 \\ 1\ 0\ 1 \\ 0\ 1\ 0 \\ 0\ 1\ 1 \\ 0\ 1\ 1 \end{bmatrix} \cdot \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 1 \\ 1 \\ 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} = \mathbf{A}_1$$

Se observa que el patrón $A_1$ se recupera correctamente.

Ahora se le presenta el patrón $B_2$ a la BAM Alfa-Beta, el proceso de recuperación es el siguiente:

Se crea el vector **F** que es una versión ruidosa, con ruido aditivo, del patrón $\mathbf{Y}_2$ ,

$$\mathbf{F} = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} \quad \text{y se le presenta a la matriz } \mathbf{V}\text{x}, \quad \mathbf{R} = \mathbf{V}x\Lambda_{\beta}\mathbf{F} = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 1 \\ 0 \end{bmatrix}$$

Realizamos la contracción $\mathbf{r} = \tau^c(\mathbf{R},n)$, entonces:

$$\mathbf{r} = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}, \text{ el cual corresponde al } 2^{\text{o}} \text{ vector } \textit{one-hot}$$

El vector **r** se le presenta a la matriz **LAx** y se realiza la siguiente operación:

$$\mathbf{LAx} \cdot \mathbf{r} = \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 0 \\ 0 & 1 & 1 \\ 1 & 1 & 0 \\ 1 & 0 & 1 \\ 1 & 0 & 1 \\ 0 & 1 & 0 \\ 0 & 1 & 1 \\ 0 & 1 & 1 \end{bmatrix} \cdot \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \\ 1 \\ 1 \\ 0 \\ 0 \\ 1 \\ 1 \\ 1 \end{bmatrix} = \mathbf{A}_2$$

Se observa que el patrón $A_2$ se recupera correctamente.

Por último, se le presenta el patrón $B_3$ a la BAM Alfa-Beta, el proceso de recuperación es el siguiente:

Se crea el vector **F** que es una versión ruidosa, con ruido aditivo, del patrón $\mathbf{Y}_3$ ,

$$\mathbf{F} = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 1 \\ 0 \\ 0 \\ 1 \\ 0 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} \quad \text{y se le presenta a la matriz } \mathbf{V}\text{x}, \quad \mathbf{R} = \mathbf{V}x\Lambda_{\beta}\mathbf{F} = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 1 \\ 0 \\ 0 \\ 1 \\ 0 \\ 1 \\ 0 \\ 0 \\ 1 \end{bmatrix}$$

Realizamos la contracción $\mathbf{r} = \tau^c(\mathbf{R},n)$, entonces:

$$\mathbf{r} = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}, \text{ el cual corresponde al 3er.vector } \textit{one-hot}$$

El vector **r** se le presenta a la matriz **LAx** y se realiza la siguiente operación:

$$\mathbf{LAx} \cdot \mathbf{r} = \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 0 \\ 0 & 1 & 1 \\ 1 & 1 & 0 \\ 1 & 0 & 1 \\ 1 & 0 & 1 \\ 0 & 1 & 0 \\ 0 & 1 & 1 \\ 0 & 1 & 1 \end{bmatrix} \cdot \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 1 \\ 0 \\ 1 \\ 1 \\ 0 \\ 1 \\ 1 \end{bmatrix} = \mathbf{A}_3$$

Se observa que el patrón $A_3$ se recupera correctamente.

En este ilustrativo ejemplo numérico se mostró que la BAM Alfa-Beta tuvo recuperación correcta de todos los patrones entrenados.

Shen y Cruz [18] utilizan los siguientes pares de patrones para entrenar su modelo de BAM que utiliza algoritmos genéticos.


Los autores encuentran que el límite superior de la energía del hiper-radio (ver Apéndice B) es

$$\hat{F} = \frac{50}{2} + 1 = 26$$

Para su simulación proponen $F^* = 5$. Con este valor encontrado, su BAM tiene un tiempo de cálculo de, aproximadamente, 15 horas.

Se entrenaron los mismos patrones en la BAM Alfa-Beta el tiempo de cálculo requerido por nuestro algoritmo es, aproximadamente, de 2 segundos.

## 5.2 Aplicación 1. Patrones fundamentales: figuras monocromáticas

La primera ventana que aparece en este programa se muestra en la figura 5.1


**Figura 5.1** Menú de opciones de la primera aplicación.

Al hacer clic sobre el botón de “Elegir Conjuntos Fundamentales” aparece la pantalla que se muestra en la figura 5.2.

Cada una de las figuras mostradas en esta pantalla se obtuvo de la página de Microsoft (www.microsoft.com) eligiendo buscar *clipart*. Originalmente las figuras se obtienen en formato *wmf*. Se utilizó el software de edición de imágenes *Advanced Batch Converter* para convertir el formato original a formato *gif*. Las dimensiones de las figuras eran de 60 x 58 pixeles; con el mismo editor de imágenes se redujo la dimensión a 50 x 50 pixeles. Este último proceso se realizó para que las 104 imágenes tuvieran cabida en la pantalla.

La elección de las asociaciones se realiza de la siguiente manera: se posiciona el ratón sobre la imagen deseada y se dan dos clic con el botón izquierdo del ratón; automáticamente aparece la imagen elegida en la parte inferior derecha de la pantalla abajo del título “Entrada”. Para elegir el patrón que se asociará a esta imagen previa, se realiza el mismo proceso. La diferencia es que la imagen elegida aparecerá en la parte superior derecha debajo del título “Salida”. De esta manera se puede crear el conjunto fundamental de hasta 52 pares de patrones.

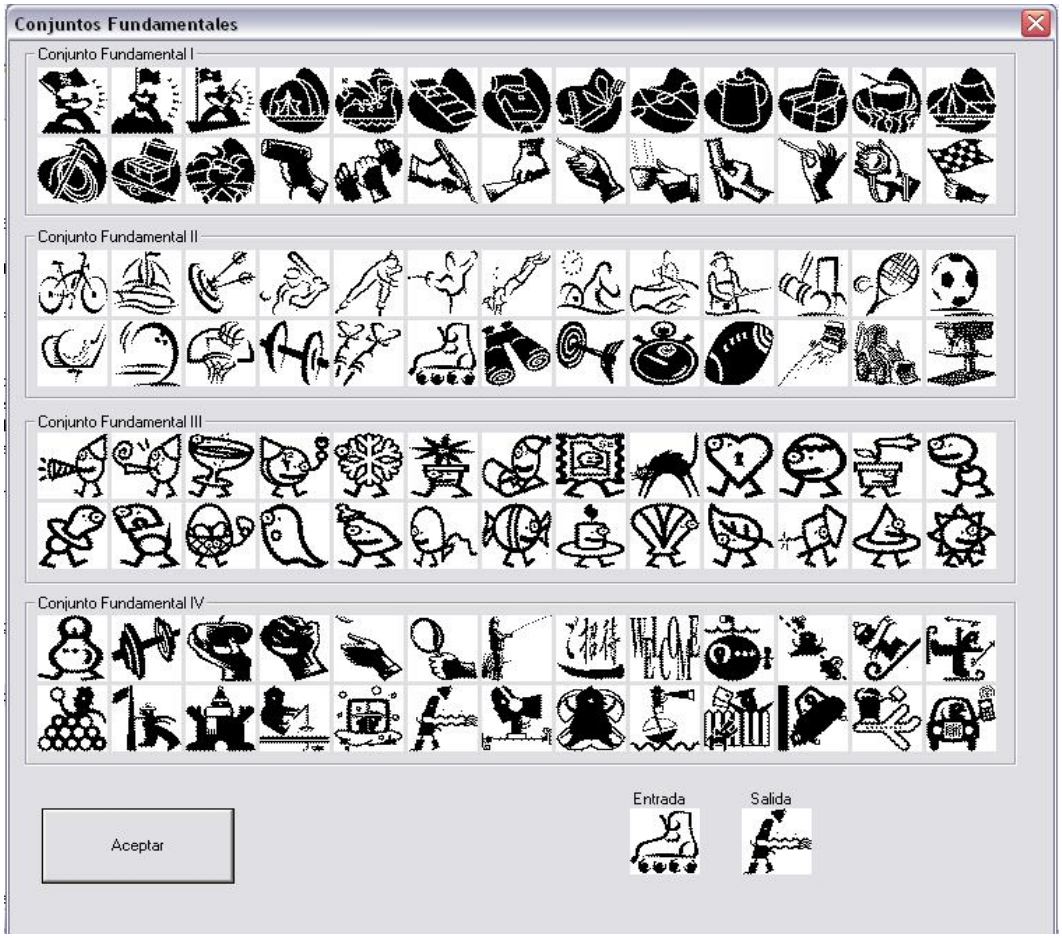



**Figura 5.2** Pantalla para la elección del conjunto fundamental para la memoria asociativa bidireccional Alfa-Beta.

Una vez que ya se creó el conjunto fundamental se hace clic en el botón de “Aceptar”, acción que nos retorna a la pantalla principal (Fig. 5.1). Se habilita el botón de Fase de Aprendizaje, el cual nos permite crear la memoria asociativa bidireccional Alfa-Beta.

Para este ejemplo, se eligieron 52 parejas de patrones. El proceso de la fase de aprendizaje se realiza en, aproximadamente, 37 segundos.

Para la ejecución del programa se utilizó una Laptop SONY® VAIO®, con un procesador Intel Pentium 4, a 2.8 GHz.

La pantalla que aparece al hacer clic sobre el botón “Fase de Recuperación” se muestra en la figura 5.3. En la parte superior se encuentran los 52 patrones, acomodados en 4 filas de 13 imágenes, que estas asociados a los 52 patrones que se muestran el parte inferior, acomodados de la misma forma.

La elección de un patrón se realiza haciendo doble clic con el botón izquierdo del ratón sobre la imagen elegida. La figura aparecerá debajo de la etiqueta “Patrón Elegido”. Automáticamente, el patrón recuperado se mostrará a la derecha de esta imagen y debajo de la etiqueta “Patrón Recuperado”. En este ejemplo se eligió un patrón del conjunto mostrado en la parte superior.

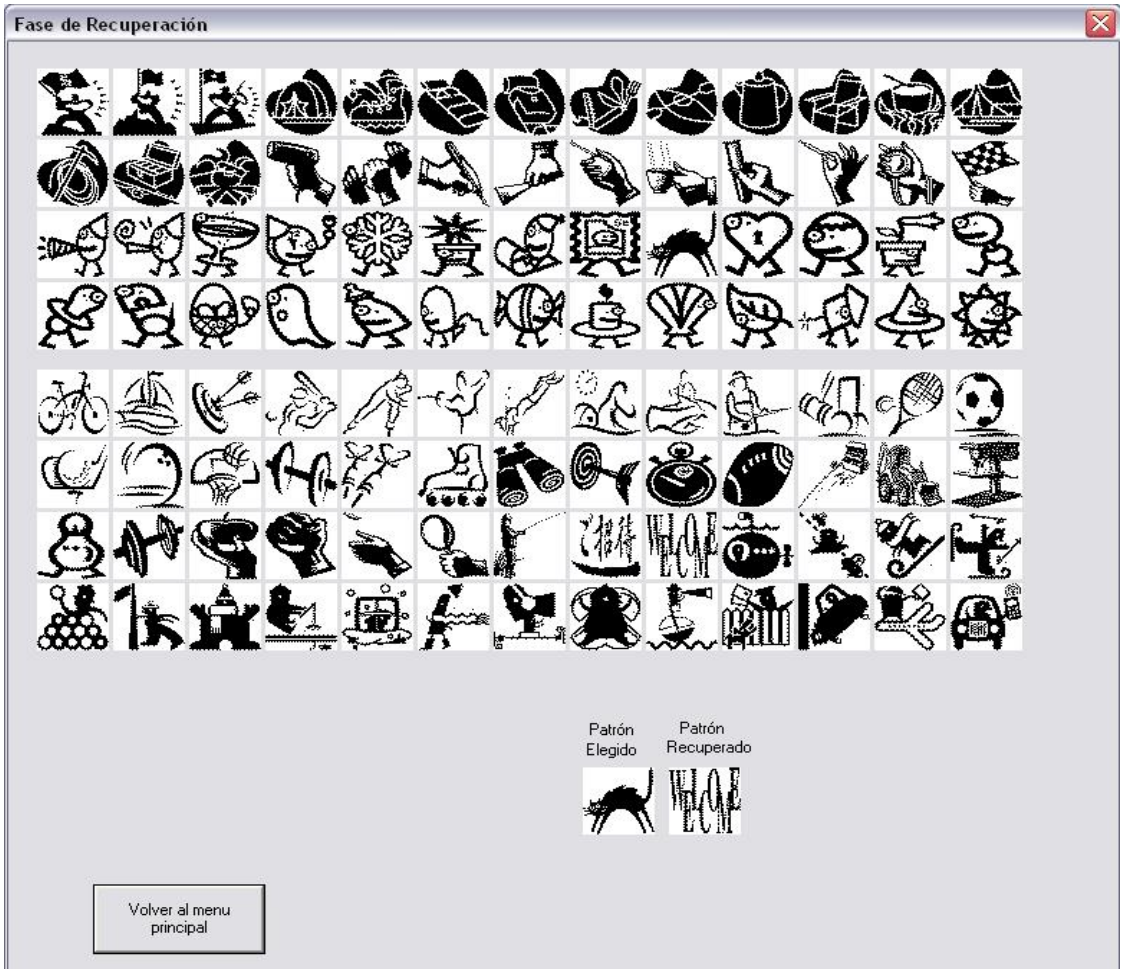



**Figura 5.3** Pantalla de la Fase de Recuperación. Haciendo doble clic sobre alguna de las imágenes se elige un patrón que aparece como “Patrón Elegido”, de manera automática se ejecuta la fase de recuperación que da como resultado otra imagen que corresponde al “Patrón Recuperado”.

Para la recuperación en el otro sentido, de igual forma, se elige una imagen del conjunto de abajo y se recupera de manera perfecta el patrón asociado (ver figura 5.4).

Todos los patrones, como era de esperarse, se recuperan de forma correcta; ya sea en un sentido como en el otro.

## 5.3 Aplicación 2. Identificador de Huellas Digitales.

En esta aplicación se asocian 40 huellas digitales a 40 números. Las huellas digitales fueron obtenidas del Fingerprint Verification Competition (FPV2000) [67]. Originalmente las imágenes tienen dimensión de 240 x 320 pixeles. Se utilizó el editor de imágenes *Advanced Batch Converter* para recortar la dimensión de las huellas, de manera que se pudieran mostrar las 40 huellas en la pantalla. La dimensión final de las imágenes es de 80 x 170 pixeles.

La pantalla del menú principal se muestra en la figura 5.5.

Al hacer clic en el botón de "Cargar Datos", se leen los archivos tanto de las huellas digitales como de los números. Además, se crea la memoria asociativa bidireccional Alfa-Beta. El proceso tarda aproximadamente 1 minuto y 34 segundos.

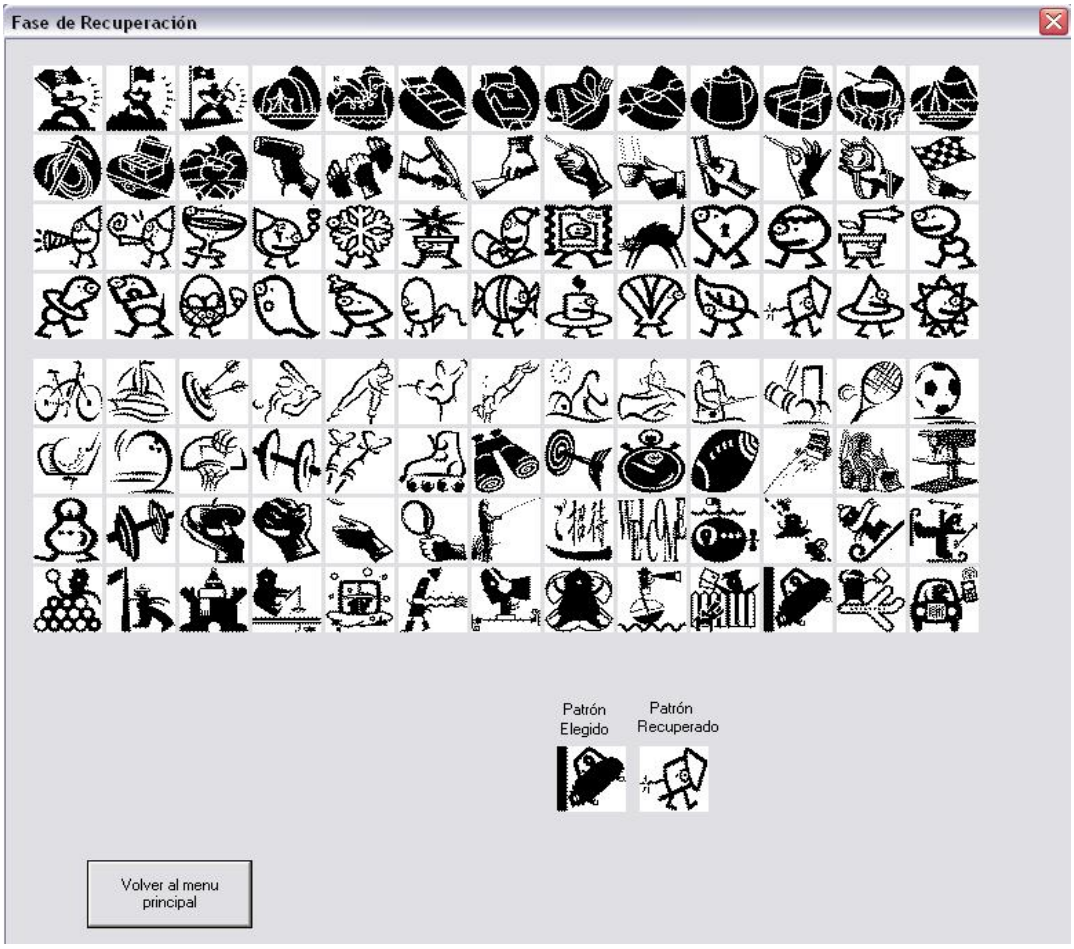



**Figura 5.4** La recuperación también se realiza hacia el otro sentido, es decir, se elige una imagen del conjunto de abajo y se recupera de manera perfecta su patrón asociado.


**Figura 5.5** Pantalla del menú principal del Identificador de Huellas Digitales
Una vez que se leyeron los archivos se procede a la fase de recuperación haciendo clic en el botón “Fase de Recuperación”. La pantalla que aparece se muestra en la figura 5.6.

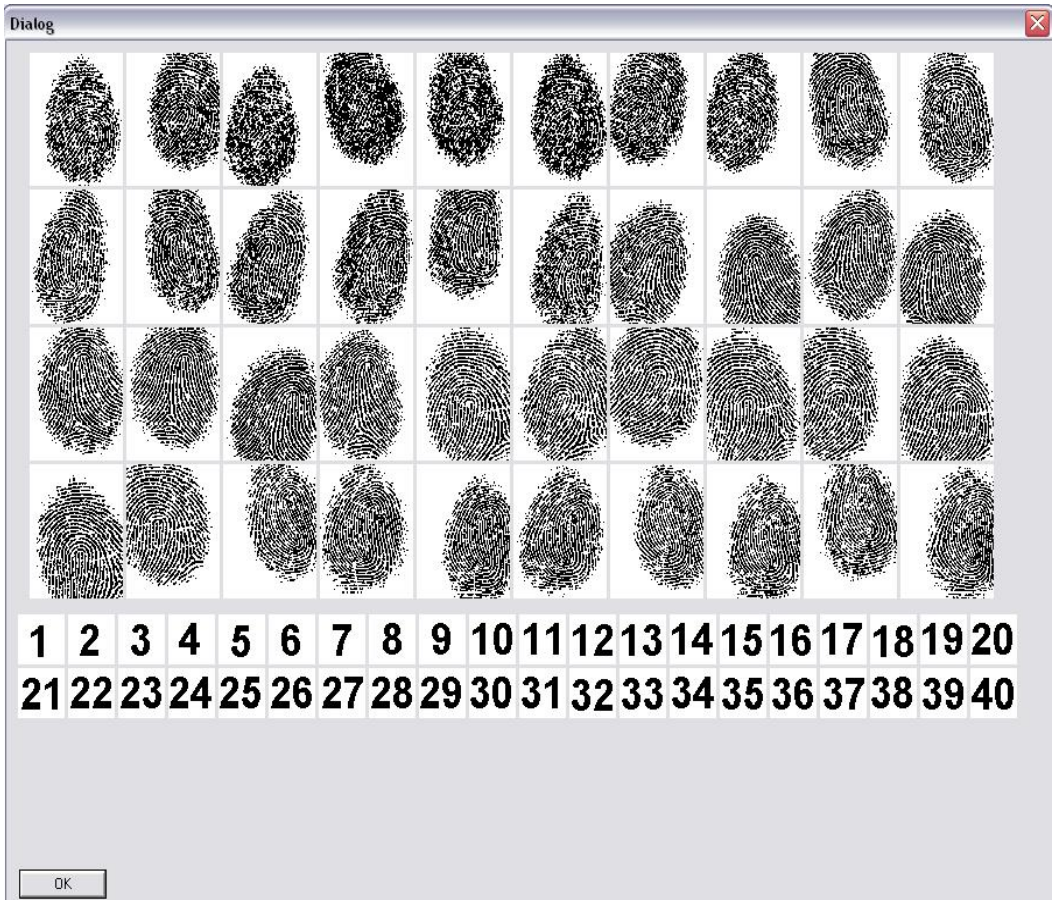



**Figura 5.6** Pantalla de la fase de recuperación

En esta pantalla se puede elegir la huella digital, haciendo doble clic con el botón izquierdo del ratón sobre la huella deseada, y entonces aparecerá, en la parte inferior derecha, la huella digital elegida y el número que le corresponde (ver figura 5.7).

La memoria asociativa bidireccional Alfa-Beta nos permite elegir un número y obtener la huella digital asociada a dicho número (ver figura 5.8).

La recuperación es correcta cuando se eligen cada una de la huellas digitales, es decir, se recuperan cada uno de los números asociados a las huellas digitales. De igual forma, se recuperan de manera correcta todas las huellas digitales correspondientes a cada uno de los números.

La memoria asociativa bidireccional Alfa-Beta permitió recuperar de manera correcta, sin ambigüedad ni condiciones, en ambas aplicaciones, para cada uno de los conjuntos fundamentales.

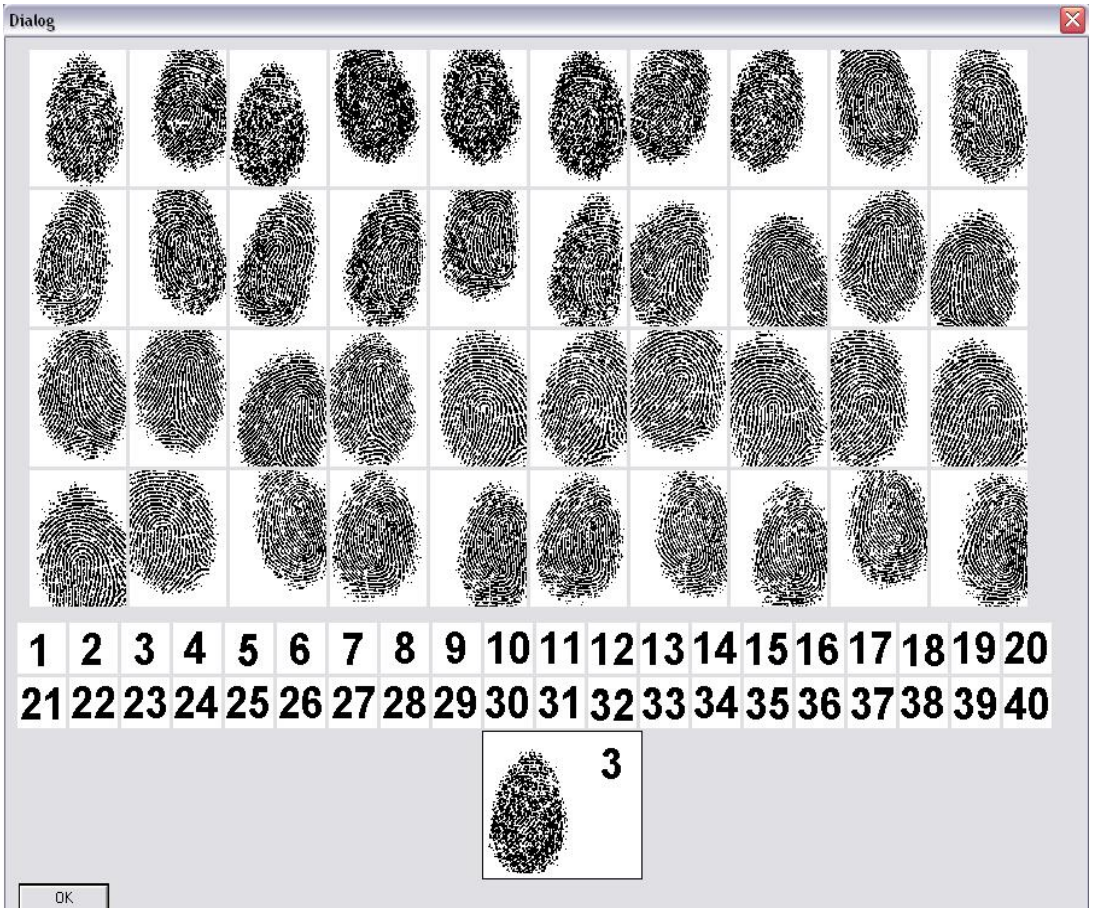



**Figura 5.7** Haciendo doble clic con el botón izquierdo del ratón se elige la huella digital y aparece, en la parte inferior derecha la huella digital elegida y su correspondiente número.

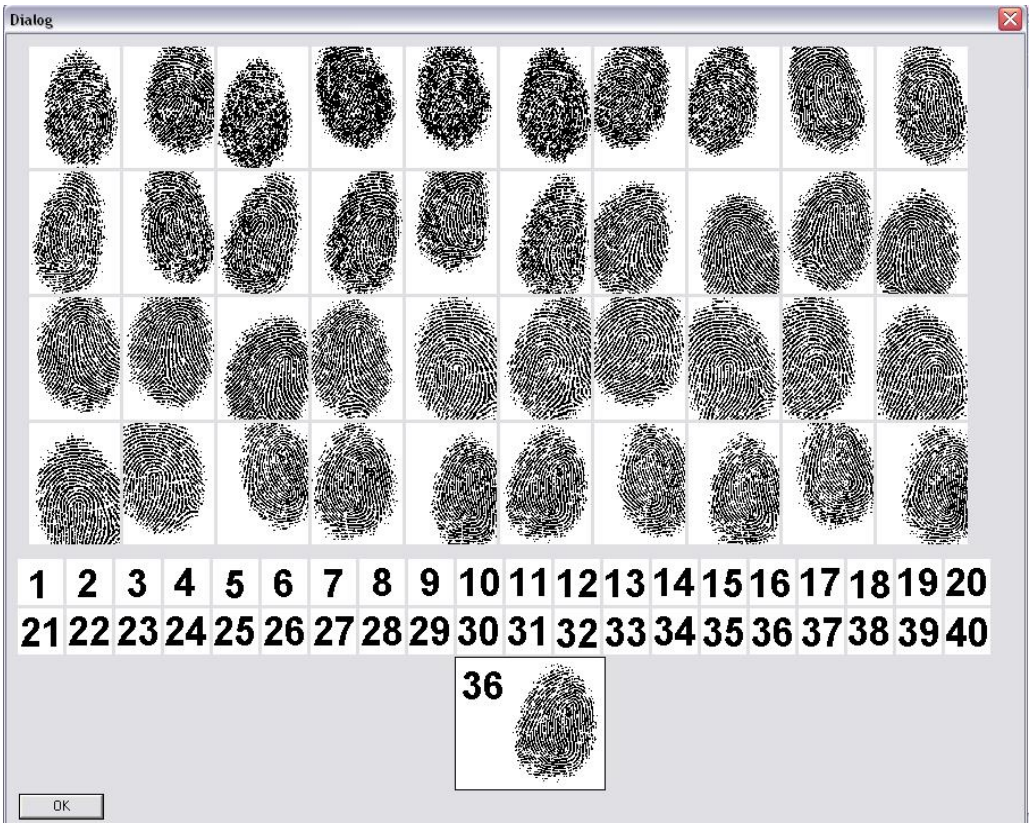



**Figura 5.8** La recuperación de patrones también se puede realizar en ambos sentidos. En este caso, se elige un número y se recupera su correspondiente huella digital.

## 5.4 Aplicación 3. Traductor Inglés-Español/Español-Inglés

En esta aplicación se utilizó el modelo de Memoria Asociativa Bidireccional Alfa-Beta para implementar un traductor inglés-español/español-inglés.

Para la fase de aprendizaje se utilizan dos archivos de texto que contienen, cada uno, 120 palabras en inglés y en español, respectivamente. Con ambos archivos se crea la Memoria Asociativa Bidireccional Alfa-Beta (Véase figura 5.9).

El proceso de aprendizaje tarda, aproximadamente, 1’ 6’’. El programa fue ejecutado en una Laptop Sony VAIO, Pentium 4 a 2.8 GHz.

Una vez creada la BAM Alfa-Beta, entonces, en la fase de recuperación, se escribe una palabra, ya sea en inglés o en español y se elige el modo de traducción. Inmediatamente aparece la palabra en el idioma correspondiente. Se puede visualizar un ejemplo en la figura 5.10. La palabra a traducir es “accuracy” y su correspondiente traducción en español es “exactitud”.


**Figura 5.9** Se crea la Memoria Asociativa Bidireccional Alfa-Beta al asociar 120 palabras en español con 120 palabras en inglés contenidas en dos archivos de texto.

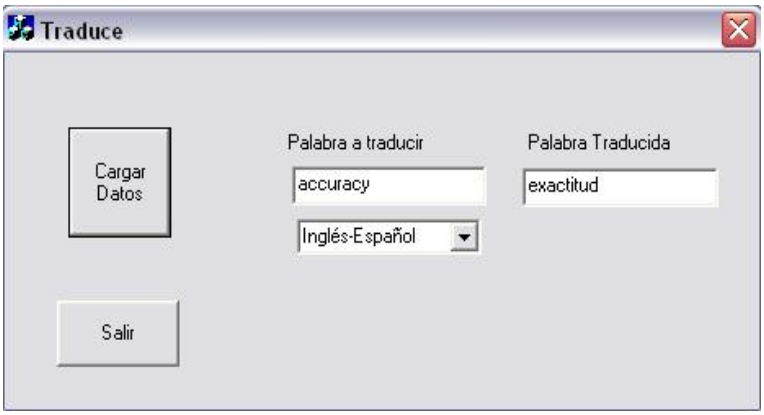



**Figura 5.10** Se escribe la palabra que se desea traducir, en este ejemplo fue “accuracy” y se elige el modo de traducción, instantáneamente aparece su correspondiente palabra en español que es “exactitud”.

El traductor presenta otras ventajas. Por ejemplo, supongamos que introducimos sólo parte de la palabra accuracy, digamos “accur”; el programa arrojará como resultado la palabra “exactitud” (Véase figura 5.11). La BAM Alfa-Beta trabaja de manera adecuada, recuperando el patrón correspondiente, debido a que la eliminación de letras se visualiza como ruido sustractivo, y el nuevo modelo propuesto en este trabajo de tesis es robusto ante este tipo de ruido.

Ahora, supongamos que en lugar de escribir una “y”, por error de escritura, se teclea una “i”. El resultado se puede observar en la figura 5.12. El patrón se recupera de manera correcta ya que, el código binario del carácter “y” es 01111001, mientras que el código binario de la “i” es 01101001, una vez más, el tipo ruido con el que se le afecta al patrón es ruido sustractivo, y la BAM Alfa-Beta recupera de manera correcta el patrón correspondiente.


**Figura 5.11** El traductor recupera de manera correcta la palabra asociada a “accuracy” aun cuando ésta no se escriba completa.

Las ventajas presentadas por el traductor resaltan las bondades de las Memorias Asociativas Alfa-Beta. Estas memorias son inmunes a cierta cantidad y tipo de ruido, propiedades que aún no se han caracterizado y que no son tema de investigación de este trabajo de tesis. Sin embargo, en la sección siguiente (Sección 5.5) se presenta una breve descripción del comportamiento de las Memorias Asociativas Bidireccionales Alfa-Beta ante patrones ruidosos.

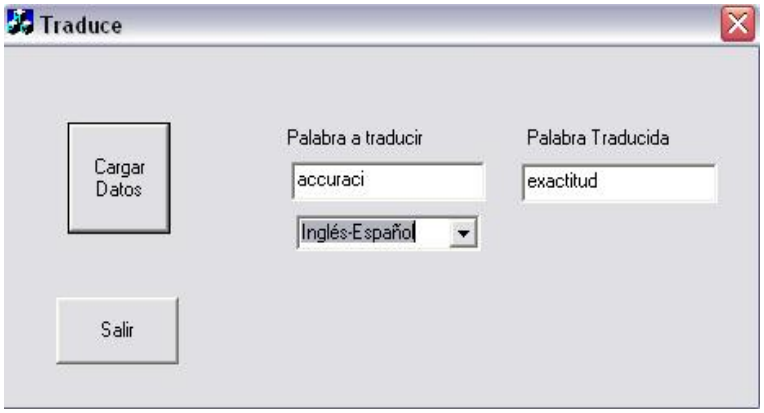



**Figura 5.12** Aun cuando exista un error de escritura y se cambie la “y” por una “i”, el programa recupera de manera perfecta la palabra “exactitud”.

Además de este tipo de pruebas, se introdujeron, completamente, las 120 palabras en inglés y se recuperaron sus respectivas traducciones en español. De igual forma, se escribieron las 120 palabras en español, y de manera correcta y sin ambigüedades, el traductor mostró las palabras en inglés correspondientes.

## 5.5 Comportamiento de la BAM Alfa-Beta ante patrones ruidosos

Se ha demostrado ya, mediante el fundamento matemático presentado en el capítulo 4 y a través de las aplicaciones mostradas en este capítulo, que el modelo de Memoria Asociativa Bidireccional recupera correctamente todos los patrones aprendidos. Sin embargo, y aunque no es parte del trabajo de investigación de esta tesis, resulta interesante conocer el comportamiento del modelo de BAM Alfa-Beta ante patrones afectados, ya sea por ruido aditivo, sustractivo o mezclado. Lo anterior, con el propósito de abrir líneas de investigación que permitan caracterizar el ruido soportado por el nuevo modelo propuesto y, de esta manera, mejorar el desempeño de la BAM Alfa-Beta ante patrones ruidosos.

Se tomaron tres imágenes del conjunto mostrado en la figura 5.4. En la figura 5.13 se indican las imágenes elegidas. A la primera de ellas se le introdujo ruido aditivo; a la segunda se le afectó con ruido sustractivo, y a la tercera se le introdujo ruido mezclado. El resultado de la recuperación de sus patrones correspondientes se muestra en la figura 5.14.

El patrón con ruido aditivo (Fig. 5.14a) se recupera correctamente, al igual que el patrón afectado por ruido sustractivo (Fig. 5.14b). Sin embargo, el patrón que tiene ruido mezclado (Fig. 5.14c) no se recupera.

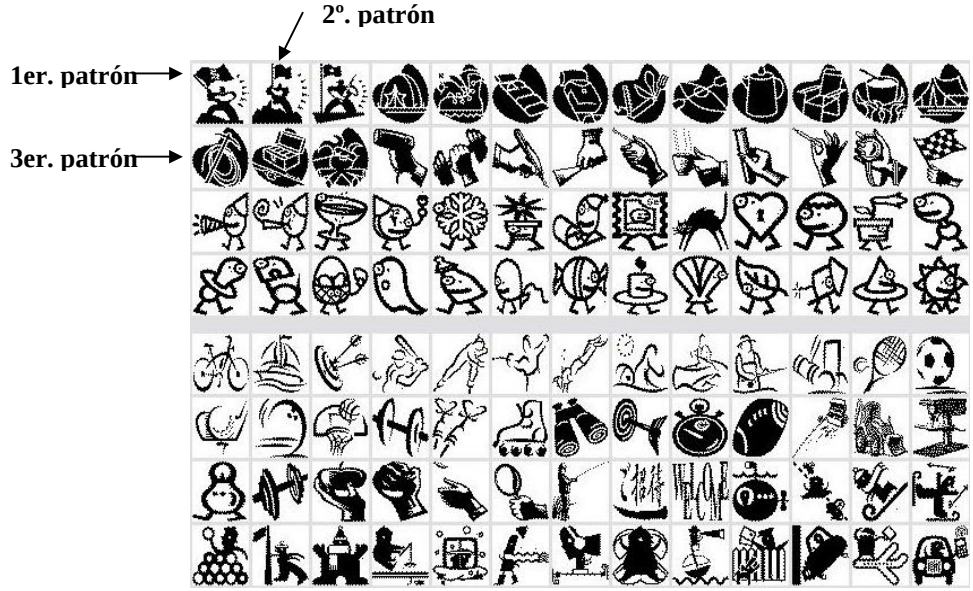



**Figura 5.13** Patrones elegidos para realizar pruebas con ruido, tomados del ejemplo mostrado en la figura 5.4.

La razón por la cual la BAM Alfa-Beta recupera de manera correcta los patrones afectados por ruido aditivo y sustractivo, es por la forma en que está construida. Las Etapas 1 y 3 constan de dos memorias autoasociativas Alfa-Beta *max* y *min*; la Alfa-Beta *max* se comporta de manera muy eficiente ante ruido aditivo, mientras que la Alfa-Beta *min* es muy resistente al ruido sustractivo. Sin embargo, las memorias autoasociativas Alfa-Beta tienen problemas de recuperación cuando se les presentan patrones con ruido mezclado.

De acuerdo con el párrafo anterior, la razón por la cual la BAM Alfa-Beta no recupera de manera correcta los patrones afectados por ruido mezclado, es que este nuevo modelo *hereda* los tipos de ruido a que son sensibles las memorias autoasociativas Alfa-Beta *max* y *min*. Lo anterior, debido a que en su arquitectura el nuevo modelo incluye ambos tipos tipos de memorias autoasociativas Alfa-Beta. En la figura 5.14 se observa un ejemplo de recuperación correcta con ruido aditivo, un ejemplo de recuperación correcta con ruido sustractivo y tres ejemplos donde la recuperación no es correcta, ante distintos niveles de ruido mezclado.

## 5.6 Recuperación correcta del conjunto fundamental completo

El fundamento matemático en el cual se sustenta el diseño y operación del modelo de las Memorias Asociativas Bidireccionales Alfa-Beta, demuestra que el nuevo modelo original propuesto en este trabajo de tesis, presenta recuperación correcta de todos los patrones del conjunto fundamental. Esta característica le permite a la Memoria Asociativa Bidireccional Alfa-Beta superar, en cuanto a la capacidad de almacenamiento, a todos los modelos anteriormente propuestos.

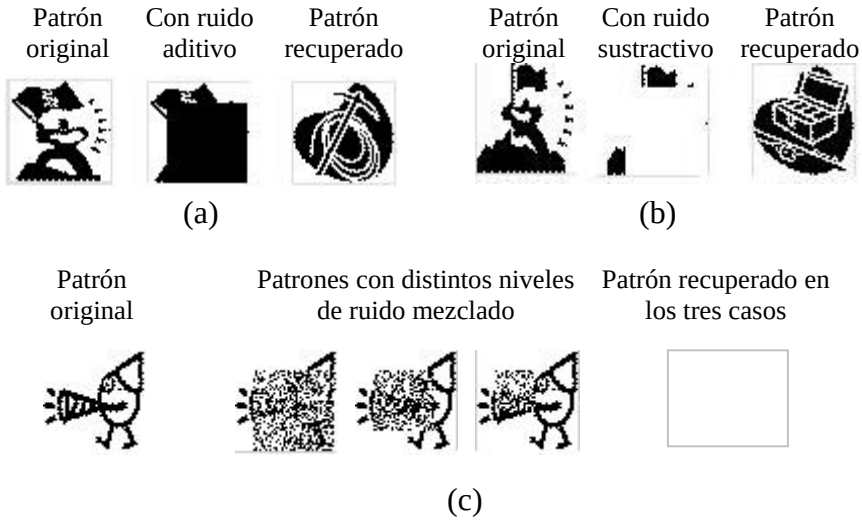



**Figura 5.14** Se muestran tres patrones afectados por (a) ruido aditivo, (b) ruido sustractivo y (c) tres distintos niveles de ruido mezclado.

En la figura 5.15 se muestran las gráficas de la capacidad de almacenamiento correspondientes a los trabajos de Wang [19], Jeng [30], Wu [58] y Zheng [64]. Se puede observar que en el caso del modelo de Wang (Fig. 5.15a), la capacidad de recuperación decae cuando el número de patrones almacenados es mayor a 17. La BAM de Jeng (Fig. 5.15b) muestra recuperación correcta hasta de 200 patrones entrenados; sin embargo, no presenta resultados más allá de esa cantidad de patrones, ni da un sustento matemático que demuestre que su modelo funciona de manera correcta para todo el conjunto fundamental. La red neuronal de tres capas de Wu (Fig. 5.15c), tiene la limitante de que los patrones

entrenados no pueden tener una distancia de Hamming mayor a 5, dado que en ese caso, la capacidad de recuperación decrece. Se puede observar que Zheng (Fig. 5.15d) sólo muestra una gráfica de la capacidad de recuperación para 10 patrones entrenados; al igual que Jeng, no presenta de manera formal argumentos que indiquen que la capacidad de recuperación de su BAM siempre es correcta.

En cada una de las gráficas mostradas en la figura 5.15, se ha dibujado una flecha indicando la capacidad de almacenamiento de las Memorias Asociativas Bidireccionales Alfa-Beta. El nuevo modelo propuesto **siempre presentará recuperación correcta de todos los patrones del conjunto fundamental**, sin condiciones previas y sin importar el número de patrones del conjunto fundamental.

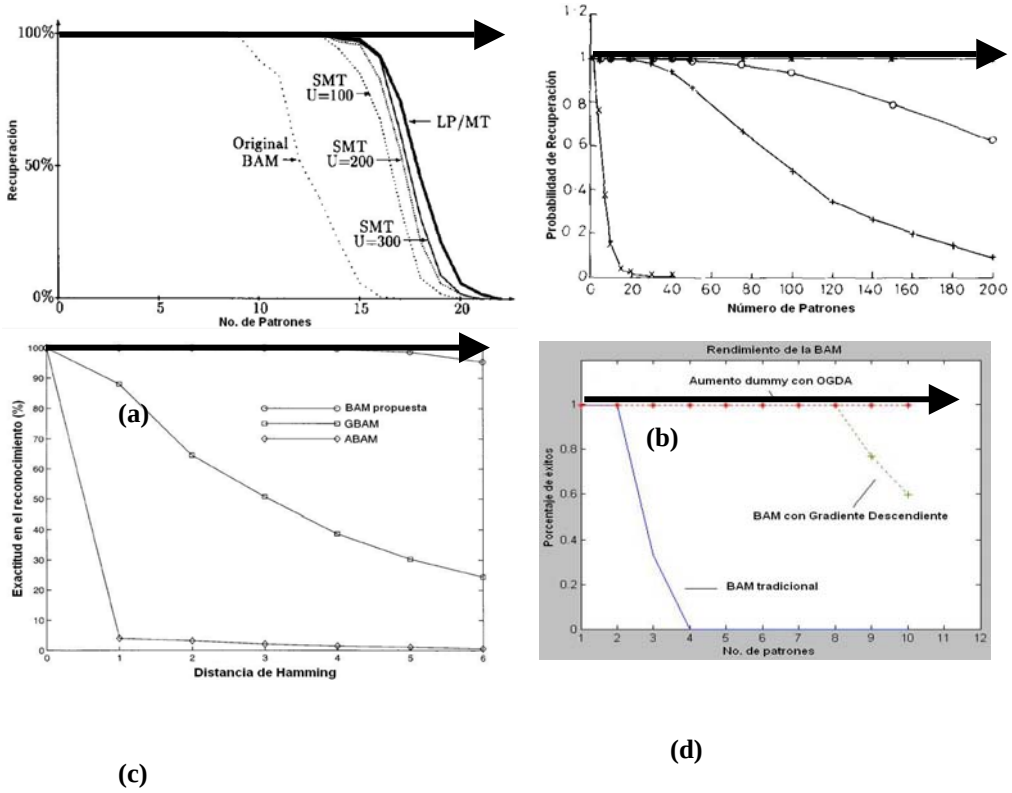



**Figura 5.15** Gráficas de la capacidad de almacenamiento de modelos de BAM. La flecha indica que la capacidad de la BAM Alfa-Beta siempre es correcta sin importar el número de patrones entrenados.

# Conclusiones y trabajo futuro

En esta sección se presentan las conclusiones obtenidas al analizar los desarrollos y resultados originales del nuevo modelo de Memoria Asociativa Bidireccional denominado Alfa-Beta, motivo central de este trabajo de tesis. Además, se hace una propuesta con una serie de posibles trabajos de investigación cuya motivación y arranque pueden derivarse a partir del nuevo modelo desarrollado y de sus aplicaciones.

## Conclusiones

1. Se ha realizado un estudio amplio del estado del arte en memorias asociativas bidireccionales. Esto ha permitido conocer las técnicas que se utilizaron a través de la evolución de los modelos de memorias asociativas bidireccionales, así como las ventajas y desventajas de cada modelo.

2. Este trabajo de tesis tiene como producto original un modelo de memoria asociativa bidireccional que no se basa en la convergencia hacia estados estables, como lo hace la mayoría de sus antecesores, sino que funciona con base en un algoritmo no iterativo.

3. Se proponen dos nuevas transformadas vectoriales: transformada de expansión y transformada de contracción.

4. Las nuevas memorias asociativas bidireccionales Alfa-Beta superan a los modelos anteriormente propuestos en la capacidad de recuperación: la BAM Alfa-Beta presenta recuperación correcta de todos los patrones de entrenamiento.

5. Los patrones de entrenamiento por la BAM Alfa-Beta no están limitados a patrones ortogonales, linealmente separables o con cierta distancia de Hamming. Las propiedades de los patrones no limita a la BAM Alfa-Beta, la cual, sin condiciones adicionales, presenta recuperación correcta de todos los patrones del conjunto fundamental.

6. La recuperación correcta de los patrones del conjunto fundamental, presentada por la BAM Alfa-Beta, se basa principalmente en la capacidad máxima de almacenamiento y aprendizaje mostradas por las memorias autoasociativas Alfa-Beta, pilar fundamental en el esquema del nuevo modelo de BAM.

7. La estructura del nuevo modelo de BAM Alfa-Beta está constituida por una memoria autoasociativa Alfa-Beta *max* y una memoria autoasociativa Alfa-Beta *min*, por lo que heredan las propiedades de las mismas.

8. La BAM Alfa-Beta recupera patrones afectados por ruido aditivo ya que las memorias autoasociativas Alfa-Beta *max* son robustas ante grandes cantidades de ruido aditivo.

9. La BAM Alfa-Beta recupera patrones afectados por ruido sustractivo ya que las memorias autoasociativas Alfa-Beta *min* son robustas ante grandes cantidades de ruido sustractivo.

10. Los patrones ruidosos que contienen ruido mezclado no son recuperados por la BAM Alfa-Beta debido a que las dos memorias autoasociativas Alfa-Beta: *max* y *min*, presentan problemas de recuperación ante este tipo de ruido.

11. La caracterización del ruido soportado por las BAM Alfa-Beta es materia de trabajos posteriores, dado que la investigación realizada en este trabajo de tesis tiene como objetivo principal la recuperación correcta de patrones del conjunto fundamental.

12. Los resultados originales de este trabajo de tesis conforman un marco de trabajo sustentado por una serie de lemas y teoremas, que permitirá a otros investigadores incursionar en esta línea de investigación.

13. La complejidad del algoritmo de las memorias asociativas bidireccionales Alfa-Beta es $O(n^2)$.

## Trabajo futuro

1. Implementar las nuevas memorias asociativas bidireccionales como clasificadores. Por ejemplo, configurando el conjunto fundamental, de modo que los patrones de salida representen clases, a la manera de los patrones de salida de la *Lernmatrix* de Steinbuch [10, 68].

2. Llevar a cabo estudios comparativos de los clasificadores, que utilicen la BAM Alfa-Beta, con los clasificadores conocidos [68, 69, 70]

3. Utilizar las memorias asociativas bidireccionales para la compresión de imágenes [71].

4. Analizar y explicar la teoría del desorden del estrés postraumático [72] mediante memorias asociativas bidireccionales.

5. Solucionar problemas de agrupamiento automático de patrones a través de memorias asociativas bidireccionales [73]

6. Realizar la representación de lattices de conceptos utilizando memorias asociativas bidireccionales [74, 75].

## Relación de publicaciones propias

- Artículo aceptado en congreso internacional cuyos proceedings serán publicados en la revista internacional ISI Lecture Notes in Computer Science de Springer-Verlag, The 13th International Conference on Neural Information Processing (ICONIP 2006): "Alpha-Beta Bidirectional Associative Memories".

- Artículo publicado en revista internacional con arbitraje (3 arbitrajes): "Alpha-Beta Bidirectional Associative Memories Based Translator". International Journal of Computer Science and Network Security, Vol.6 No. 5A, May 2006.

- Artículo enviado a: 11th Iberoamerican Congress on Pattern Recognition CIARP 2006: Fingerprint Verification using Alpha-Beta Bidirectional Associative Memories.

- Artículo aceptado en el 5th. Mexican International Conference on Artificial Intelligence MICAI 2006: "Complexity of Alpha-Beta Bidirectional Associative Memories".

- Artículo aceptado en congreso internacional cuyos proceedings serán publicados en la revista internacional ISI Lecture Notes in Computer Science de Springer-Verlag, The 21st International Symposium on Computer and Information Sciences Program Committee (ISCIS 2006): "A New Model of BAM: Alpha-Beta Bidirectional Associative Memories".

# Apéndice A

# Simbología

| | |
|---|---|
| $\mathbf{M}$ | memoria asociativa |
| $\mathbf{x}$ | vector columna que corresponde a un patrón de entrada |
| $\mathbf{y}$ | vector columna que corresponde a un patrón de salida |
| $(\mathbf{x}, \mathbf{y})$ | asociación de un patrón de entrada con uno de salida |
| $(\mathbf{x}^k, \mathbf{y}^k)$ | asociación de la $k$-ésima pareja de patrones |
| $\{(\mathbf{x}^\mu, \mathbf{y}^\mu) \mid \mu = 1, 2, ..., p\}$ | conjunto fundamental |
| $A$ | conjunto al que pertenecen los vectores **x** y **y** |
| $B$ | conjunto al que pertenecen las entradas de la matriz **M** |
| $p$ | número de parejas del conjunto fundamental |
| $n$ | dimensión de los patrones de entrada |
| $m$ | dimensión de los patrones de salida |
| $\forall$ | cuantificador universal |
| $\in$ | pertenencia de un elemento a un conjunto |
| $\exists$ | cuantificador existencial |
| $y_k^\mu$ | $k$-ésima componente del vector columna $\mathbf{y}^\mu$ |
| $\times$ | producto cruz (entre conjuntos) |
| $m_{ij}$ | $ij$-ésima componente de la matriz **M** |
| $\Delta m_{ij}$ | incremento en $m_{ij}$ |
| $\cdot$ | producto usual entre vectores o matrices |
| $\vee$ | operador máximo |
| $(\mathbf{x}^\mu)^t$ | Transpuesto del vector $\mathbf{x}^\mu$ |
| $\delta_{ij}$ | delta de Kronecker (afecta a los índices $i$ y $j$ ) |
| $\mathbf{I}$ | matriz identidad |
| $\tilde{\mathbf{x}}$ | versión alterada del patrón fundamental **x** |
| $\mathbf{W}$ | memoria asociativa morfológica *min* |
| $\nabla$ | producto máximo |
| $\Delta$ | producto mínimo |
| $\wedge$ | operador mínimo |
| $\alpha$ y $\beta$ | operadores en que se basan las memorias Alfa-Beta |
| $\alpha(x, y)$ | operación binaria $\alpha$ con argumentos $x$ y $y$ |
| $\beta(x, y)$ | operación binaria $\beta$ con argumentos $x$ y $y$ |
| A y B | operadores en que se basan las memorias Media |
| **med** | operador media |
| $\nabla_\alpha$ | operador $\alpha$ *max* |
| $\nabla_\beta$ | operador $\beta$ *max* |
| $\Delta_\alpha$ | operador $\alpha$ *min* |

| | |
|---|---|
| $\boldsymbol{\Delta}_{\beta}$ | operador $\beta$ *min* |
| $\otimes$ | símbolo que representa a las dos operaciones $\nabla_{\alpha}$ y $\Delta_{\alpha}$ |
| $\mathbf{V}$ | memorias asociativas Alfa-Beta tipo *max* |
| $\boldsymbol{\Lambda}$ | memorias asociativas Alfa-Beta tipo *min* |
| $\mathbf{Z}^{+}$ | conjunto de los números enteros positivos |
| $\mathbf{h}^{k}$ | *k*-ésimo vector *one-hot* |
| $\bar{\mathbf{h}}^{k}$ | *k*-ésimo vector *zero-hot* |
| $\tau^{e}$ | transformada vectorial de expansión |
| $\tau^{c}$ | transformada vectorial de contracción |
| $\bar{s}$ | Vector negado del vector **s** |
| **LAy** | *Linear Associator* modificado para **y** |

# Apéndice B

# Memorias Asociativas Bidireccionales a través del tiempo (detalle)

En este apéndice se presentan, en detalle, las 38 Memorias Asociativas Bidireccionales más importantes a través del tiempo, que se mencionan en la tabla 2.3.1. Además se describen las BAM con retardos, modelos que intentan mejorar la capacidad de recuperación.

## *Hecht-Nielsen (1988)*

En 1988, Hecht-Nielsen y Haines del Departamento de Ingeniería Eléctrica y Computación de la Universidad de California, hacen una generalización de la BAM de Kosko, la cual nombraron Memoria Asociativa Bidireccional No Homogénea [16]. En el trabajo de Kosko [14], los umbrales utilizados para cambiar de un estado a otro, tanto de los vectores de entrada como de salida, se fijaban a cero. En la BAM No Homogénea (BAM-NH) los umbrales pueden tener un valor diferente de cero y, además, los umbrales para los vectores de entrada difieren de los de los vectores de salida.

Para la codificación de la BAM, se dan $L$ pares de vectores fila, ($\mathbf{x}_1$, $\mathbf{y}_1$), ($\mathbf{x}_2$, $\mathbf{y}_2$), ..., ($\mathbf{x}_k$, $\mathbf{y}_k$), ..., ($\mathbf{x}_L$, $\mathbf{y}_L$) que se codifican en una matriz ponderada mediante la suma de los productos externos de los vectores, $x_k^T y_k$ . La matriz ponderada **W** es:

$$\mathbf{W} = \mathbf{x}_1^T \mathbf{y}_1 + \mathbf{x}_2^T \mathbf{y}_2 + ... + \mathbf{x}_L^T \mathbf{y}_L \tag{B.1}$$

Dado que $\mathbf{x} \in \{-1, +1\}^n$ y $\mathbf{y} \in \{-1, +1\}^m$, la BAM (no-homogénea) se define mediante las ecuaciones de la función de transferencia:

$$x_i^{new} = \begin{cases} 1 & \text{si } \sum_{j=1}^{m} w_{ij} y_j^{old} > S_i \\ x_i^{old} & \text{si } \sum_{j=1}^{m} w_{ij} y_j^{old} = S_i \\ -1 & \text{si } \sum_{j=1}^{m} w_{ij} y_j^{old} < S_i \end{cases} \tag{B.2}$$

y

$$y_j^{new} = \begin{cases} 1 & \text{si } \sum_{i=1}^{n} w_{ij} x_i^{old} > T_j \\ y_j^{old} & \text{si } \sum_{i=1}^{n} w_{ij} x_i^{old} = T_j \\ -1 & \text{si } \sum_{i=1}^{n} w_{ij} x_i^{old} < T_j \end{cases} \quad \text{(B.3)}$$

Donde $S_i$ es el umbral para el *i*-ésimo elemento $x_i$ de **x** y $T_j$ es el umbral para el *j*-ésimo elemento de $y_j$ de y.

La energía o función de Lyapunov para la BAM no-homogénea está dada por:

$$E(\mathbf{x}, \mathbf{y}) = -\mathbf{x}\mathbf{W}\mathbf{y}^T + \mathbf{x}\mathbf{S}^T + \mathbf{y}\mathbf{T}^T \quad \text{(B.4)}$$

donde $\mathbf{S} = (S_1, S_2, \ldots, S_n)$ y $T = (T_1, T_2, \ldots, T_n)$.

Al igual que en la BAM homogénea, la función de energía de la BAM-NH decrece si y sólo si el estado de la red cambia.

En este artículo se especifica que la BAM-NH tiene una capacidad mayor de almacenamiento que la BAM original. La BAM-NH tiene una capacidad de $(0.68)n^2 / [(\log_2 n)+4]^2$, siendo superior a la BAM de Kosko.

De acuerdo con los autores, aunque la BAM-NH es mejor que la BAM, todavía está lejos de ser una buena memoria asociativa. El primer problema que presenta es que para lograr almacenar una gran cantidad de información, es necesario codificar esta información en vectores con alta dimensionalidad de manera que los datos se encuentren muy esparcidos, para evitar que los datos de los vectores se parezcan.

El segundo problema que presenta la BAM-NH es el de la convergencia. En este trabajo se especifica que el algoritmo siempre converge, pero se sigue teniendo un proceso iterativo como en el caso de Kosko.

## *Wang Y. (1989)*

Wang, Cruz y Mulligan [28] del Departamento de Ingeniería Eléctrica de la Universidad de California en Irvine, en 1989 utilizaron un concepto nuevo de codificación identificado como Entrenamiento Múltiple (*Multiple Training*).

Kosko [14] afirma que su BAM converge a algún patrón (o estado estable) debido a que cada iteración que ejecuta el algoritmo, se decrementa la energía *E* y que, por consiguiente, se estabilizará en un mínimo local de energía. Los patrones recuperados serán aquellos cuya energía sea un mínimo local. La ecuación propuesta por Kosko para el cálculo de la energía es

$$E(A, B) = -A\mathbf{M}B^T \quad \text{(B.5)}$$

A partir de esto, en este trabajo se afirma que si la energía $E$, que es evaluada utilizando la asociación ($\alpha$, $\beta$), la cual pertenece al conjunto fundamental, no constituye un mínimo local, entonces no se tendrá una recuperación exitosa. Por otro lado, si se produce un mínimo local cuando los estados iniciales pertenecen al conjunto fundamental, entonces sí se podrá recuperar el patrón.

Para asegurar que se puede recuperar un par ($A_i$,$B_i$) en particular cuando se inicializa con $\alpha$, se requiere que el patrón fundamental sea un mínimo local de la función $E$ definida en la ecuación (2.3.11). Esto implica que cuando $\alpha = A_i$ y $\beta = B_i$ se sustituyan en la ecuación, el valor obtenido para $E$ sea menor que cualquier punto en la vecindad de ($A_i$, $B_i$).

Para el entrenamiento múltiple de orden $q$ dirigido a recuperar el patrón ($A_i$, $B_i$) se aumenta la matriz **M** (la memoria asociativa) con la matriz **P** que está definida como:

$$\mathbf{P} = (q-1)X_i^T Y_i \tag{B.6}$$

donde $X_i$ y $Y_i$ son la representación bipolar de $A_i$ y $B_i$, respectivamente.

El nuevo valor de la energía evaluada en el punto ($A_i$, $B_i$) se convierte en:

$$E'(A_i, B_i) = -A_i \mathbf{M} B_i^T - (q-1)A_i X_i^T Y_i B_i^T \tag{B.7}$$

La multiplicación de esas asociaciones por el factor ($q$-1) (a lo que se le llama entrenamiento) lo que hace es aumentar el número de patrones que son iguales a esa asociación. Al aumentar el número de patrones de una asociación, entonces la energía resultante se hace más pequeña.

Al lograr que la energía de esa asociación sea un mínimo local, entonces se asegura que la recuperación de ese patrón será exitosa.

El algoritmo de la BAM-MT asegura que siempre es posible recuperar al menos uno de los patrones entrenados, aunque las condiciones suficientes y necesarias (si las hay) para una recuperación perfecta, no son conocidas. Esto es, no se ha determinado el número de repeticiones ($q$-1), para un par deseado, que asegure la recuperación de dicho patrón.

Para la comparación entre la BAM-MT y la BAM de Kosko se realizaron simulaciones en donde se generaron 200 conjuntos de prueba. Para cada conjunto, los parámetros generados aleatoriamente fueron el número de patrones $N$, donde $2 \leq N \leq 16$; la dimensión de los patrones de entrada, $n$ , para $4 \leq n \leq 256$; y la dimensión de los patrones de salida, $p$, para $4 \leq p \leq 256$, con $N \leq \sqrt{\min(n, p)}$ .

A cada conjunto de datos se le llama evento. Si todos los pares de ese conjunto son recuperados, se declara un éxito, de otra forma, es una falla.

Utilizando el método de Kosko, en el modo binario, se obtuvieron el 66% de éxitos, y en el modo bipolar el 90%. Utilizando la BAM-MT los porcentajes de los éxitos fueron del 77.5% en el modo binario y el 98.5 % en el modo bipolar.

## *Tai (1989)*

Debido a que las redes neuronales de alto orden no lineales mostraban un mejoramiento impresionante en la capacidad de la memoria y en la habilidad de corrección de errores en comparación con los modelos de memoria lineales, Tai *et al* [29] propusieron una BAM utilizando una correlación de alto orden: *High-Order Bidirectional Associative Memory* (HOBAM).

Para esta memoria, un ciclo en el proceso de recuperación para un modelo de *k*-ésimo orden se puede formular como:

$$Y = \text{sign}\left( \sum_{\alpha=1}^{M} Y_\alpha \left[ X'_\alpha X \right]^k \right) \tag{B.8}$$

$$X^1 = \text{sign}\left( \sum_{\alpha=1}^{M} X_\alpha \left[ Y'_\alpha Y \right]^k \right) \tag{B.9}$$

donde sgn($x$) = 1 si $x > 0$ y sgn($x$)=-1 de otra forma.

Se demuestra que cuanto más grande es el orden, el rendimiento es mayor. Las consecuencias de lo anterior se reflejan en una mayor capacidad de memoria que la BAM de Kosko [14].

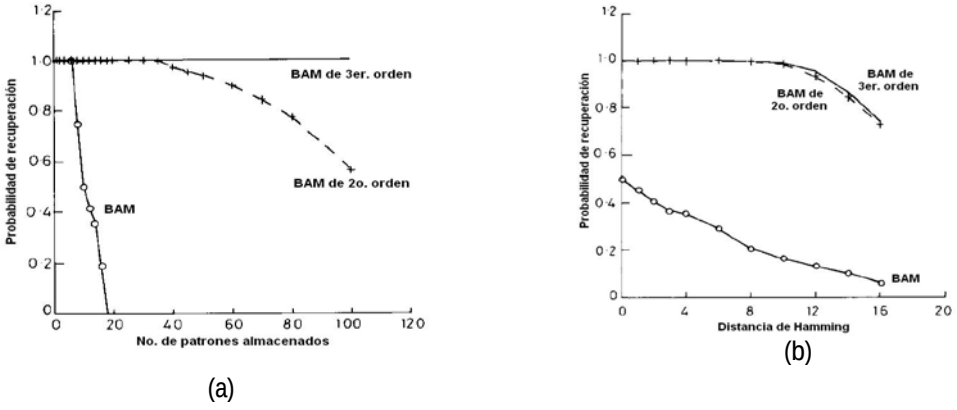



**Figura B.1** (a) Gráfica de comparación de la capacidad de almacenamiento permitida por la BAM de Kosko, HOBAM de 2º orden y la HOBAM de 3er orden. (b) Gráfica de la capacidad de corrección de errores.

De la figura B.1 se puede observar que la capacidad de la HOBAM es mejor que la BAM. Además, se demuestra que entre más grande es el orden mejor capacidad se tiene al

incrementar el número de patrones fundamentales, de la misma forma si el orden es mayor, la capacidad de corrección de errores aumenta.

## *Jeng (1990)*

Debido a que las memorias direccionables por contenido con no linealidad exponencial tienen una capacidad de almacenamiento exponencialmente escalable con el número de bits y una gran capacidad en la corrección de errores, Jeng *et al* [30], del Departamento de Ingeniería Eléctrica de la Universidad de Taiwán, propusieron una nueva memoria asociativa basada en la BAM de Kosko [14] con una no linealidad exponencial.

Se asume que hay $m$ pares de datos alamacenados $(X^i, Y^i)$, $i = 1, 2, ..., m$, donde $X^i \in \{-1, 1\}^n$ y $Y^i \in \{-1, 1\}^p$.

El proceso bidireccional de recuperación de la EBAM (*Exponential Bidirectional Associative Memory*) en un ciclo es:

$$Y = f(X) = \text{sgn}\left\{\sum_{i}^{m} Y^i \alpha^{\langle X^i, X\rangle}\right\} \tag{B.10}$$

$$X' = g(Y) = \text{sgn}\left\{\sum_{i=1}^{m} X^i \alpha^{\langle Y^i, Y\rangle}\right\} \tag{B.11}$$

donde <**A**, **B**> el producto interno de los vectores **A** y **B**, $\alpha$ es un número mayor que la unidad, y sgn ($x$) = 1 si $x \geq 0$ y sgn($x$) = -1 de otra forma.

Para una $\alpha > 1$, la no linealidad exponencial incrementa la similaridad de una entrada dada con un patrón del conjunto fundamental y reduce el *crosstalk*.

Se realizaron simulaciones para examinar la capacidad de almacenamiento de la EBAM y la HOBAM [29] de 3º y 4º orden. En la siguiente figura se muestra la capacidad de almacenamiento con $n = m = 16$. Los resultados se obtuvieron de un conjunto de 100 asociaciones generadas aleatoriamente.

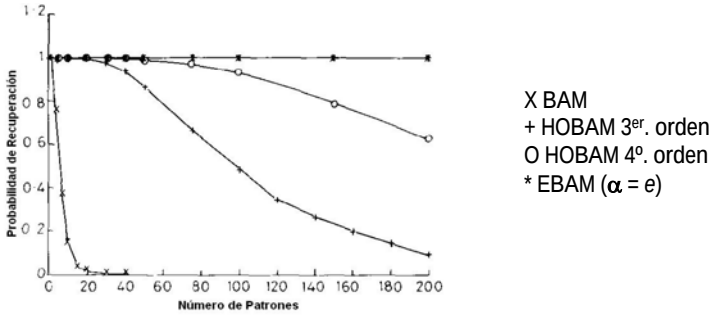



**Figura B.2** Gráfica de comparación de las capacidades de almacenamiento de la BAM, HOBAM 3er orden, HOBAM 4º orden y EBAM con $\alpha = e$.

De la figura B.2 se puede observar como la capacidad de la EBAM es mayor que la HOBAM (de 3º y 4º orden), y ni que decir de la BAM, cuya capacidad está muy por debajo de la EBAM y HOBAM.

En la figura B.3 se muestra la capacidad de corrección de errores.

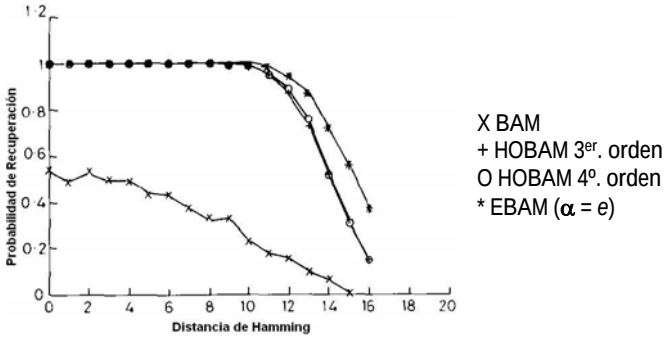



**Figura B.3** Gráfica de comparación de la capacidad de corrección de errores de la BAM, HOBAM (3er. y 4º orden) y la EBAM.

## *Wang Y. (1990)*

Este trabajo [31] es la continuación de [28] de los mismos autores, Wang *et al*, del Departamento de Ingeniería Eléctrica y de Computación de la Universidad de California, quienes aplicaron dos estrategias para lograr una mayor recuperación de patrones. La primera es la del entrenamiento múltiple [28] que permite recuperar, al menos, el patrón que se entrenó; mientras que la adición de elementos *dummies*, a cada uno de los patrones del conjunto fundamental, optimiza la recuperación de todos los pares, siempre y cuando sea posible adherir elementos *dummy* a esos pares.

Existen $N$ pares entrenados

$$\{(A_1, B_1), (A_2, B_2), ..., (A_i, B_i), \ldots, (A_N, B_N)\}$$

donde

$$A_i = (a_{i1}, a_{i2}, \ldots, a_{in})$$
$$B_i = (b_{i1}, b_{i2}, \ldots, b_{ip})$$

Se forma la matriz de correlación

$$M = \sum_{i=1}^{N} X_i^T Y_i \qquad \text{(B.12)}$$

donde $X_i(Y_i)$ es el modo bipolar de $A_i(B_i)$.

El método del aumento *dummy* se describe a continuación.

Se define un conjunto de pares entrenados como libres de ruido, si

$$\sum_{j\neq i}\left(A_i X_j^T\right)Y_j = 0 \quad \text{y} \quad \sum_{j\neq i}\left(B_i Y_j^T\right)X_j = 0 \quad \text{para toda } i.$$

También se define un conjunto de pares como estrictamente libres de ruido, si $A_i X_j^T = 0$ y $B_i Y_j^T = 0$ para todo *i*, *j*, $i \neq j$. Obviamente un conjunto estrictamente libre de ruido es un conjunto libre de ruido. Se sabe que:

$$A_i M = A_i X_i^T Y_i + \sum_{j\neq i} A_i X_j^T Y_j \tag{B.13}$$

Por lo tanto, para pares libres de ruido, $A_i M = A_i X_i^T Y_i$.

$\phi(A_i M) = \phi(sY_i) = B_i$, donde *s* es un entero positivo. Similarmente, $\phi(B_i M^T) = A_i$. Por lo tanto, para un conjunto libre de ruido, cualquier par entrenado puede ser recuperado.

Considérese *p* pares entrenados $\{A_i, B_i\}_{i=1}^N$ y un conjunto estrictamente libre de ruido de *N* elementos, $\{\mathbf{D}_i\}_{i=1}^N$, donde $\mathbf{D}_i = (d_{i1},\ldots,d_{im})$.

Entonces se construye un nuevo conjunto de patrones fundamentales:

$$\left\{\left[A_i \middle| \mathbf{D}_i \middle| \cdots \middle| \mathbf{D}_i\right], \left[B_i \middle| \mathbf{D}_i \middle| \cdots \middle| \mathbf{D}_i\right]\right\}_{i=1}^N$$

Se agregan *K* $\mathbf{D}_i$'s a $A_i$ y *K*' $\mathbf{D}_i$´s a $B_i$.

El método del aumento *dummy* requiere que el usuario genere los elementos *dummies* durante la fase de aprendizaje de la BAM. Se requiere que también se generen los elementos *dummies* para las condiciones iniciales durante la fase de recuperación.

El método del entrenamiento múltiple garantiza que uno de los pares entrenados siempre será recuperado. Sin embargo, no existe la garantía de que todos los pares se recuperen. La estrategia del aumento *dummy* garantiza que todos los pares entrenados sean recuperados.

De la figura B.4 se puede observar que el método de entrenamiento múltiple es mejor que la BAM de Kosko en ambos modos: binario y bipolar.

## *Wang Y. (1990)*

En este trabajo, Wang *et al* [35] especifican el número mínimo de veces que se tiene que entrenar a una asociación para garantizar la recuperación de ese par. La idea se basa en obtener un mínimo local de energía sin que éste sea igual al mínimo local del vecino (patrón) localizado a cierta distancia de Hamming.

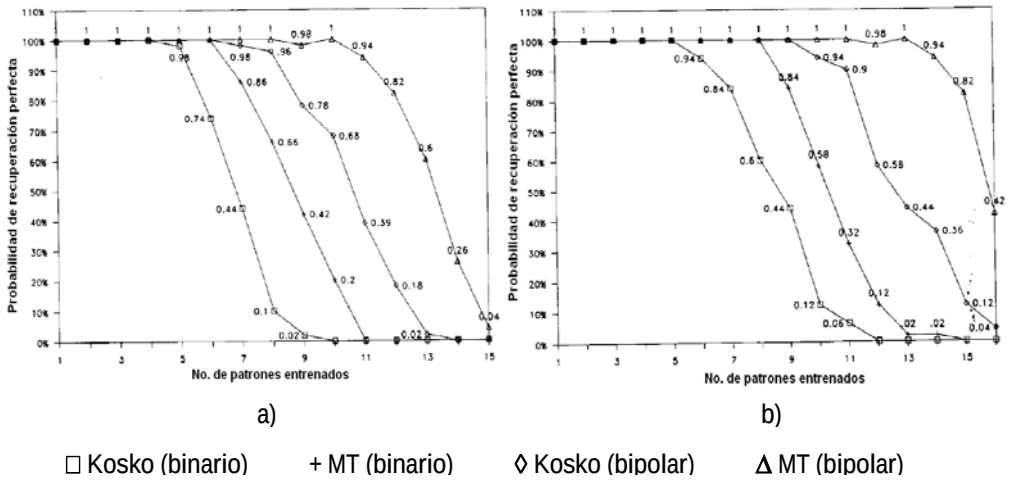



**Figura B.4** Pruebas de capacidad de almacenamiento (a) para $n = p = 100$, (b) para $n = p = 225$

Dados $N$ pares de datos $\{P_i = (A_i, B_i) | \ i = 1, ..., N\}$, donde

$$A_i = (a_{i1}, a_{i2}, ..., a_{in}),$$
$$B_i = (b_{i1}, b_{i2}, ..., b_{ip}),$$

Se define:

$\mathbf{P}_i' = (A_i', B_i')$ como el vecino a una distancia de Hamming de1 de $(A_i, B_i)$.
$E_i = -A_i \mathbf{M} B_i^T$ es la energía de $\mathbf{P}_i$ para la matriz $\mathbf{M}$.
$E_{oi} = -A_i \mathbf{M}_o B_i^T$, $E_i' = -A_i' \mathbf{M} B_i'^T$ es la energía de $\mathbf{P}_i'$ para la matriz $\mathbf{M}$.
$E_{oi}' = -A_i' \mathbf{M}_o B_i'^T$, $\varepsilon_i = E_i - E_i'$ es la diferencia de energía de $\mathbf{P}_i$ y $\mathbf{P}_i'$ para $\mathbf{M}$.
$\varepsilon_{oi} = E_{oi} - E_{oi}'$.

$(A_i, B_i)$ no pueden ser recuperados si y sólo si $\varepsilon_i > 0$ para algunos de los vecinos de $(A_i, B_i)$. Por lo tanto, se desea agregar una matriz $(q - 1)\ \mathbf{X}_i^T \mathbf{Y}_i$, donde $q \geq 1$, a $\mathbf{M}_o$ para formar una $\mathbf{M} = (q - 1)\ \mathbf{X}_i^T \mathbf{Y}_i + \mathbf{M}_o$ que resulta en una nueva $\varepsilon_i \leq 0$; digamos:

$$-A_i \mathbf{M} B_i^T - \left(-A_i' \mathbf{M} B_i'^T\right) = (q-1) * \left(-A_i X_i^T Y_i B_i^T + -A_i' X_i^T Y_i B_i'^T\right) + \varepsilon_{oi} \leq 0 \qquad \text{(B.14)}$$

Se define $\eta = -A_i X_i^T Y_i B_i^T - A_i' X_i' Y_i B_i'^T$.

Entonces la ecuación B.14 se convierte en

$$q \geq \frac{\varepsilon_{oi}}{\eta} + 1 \quad \text{si } \eta > 0 \qquad \text{(B.15)}$$

En el caso bipolar

Si el bit diferente se localiza en los patrones de entrada, entonces $\eta = 2p$ (B.16)

Si el bit diferente se localiza en los patrones de salida, entonces $\eta = 2n$ (B.17)

Debido a que *p* y *n* son positivos, entonces $\eta$ es positivo y se satisface la ecuación (B.15).

Se define:

$\varepsilon_{oi}^{A} = \max\left(E_{oi} - E_{oi}\right)$ de entre todos los vecinos de $\mathbf{P}_i$ de los patrones de entrada; y
$\varepsilon_{oi}^{B} = \max\left(E_{oi} - E_{oi}^{'}\right)$ de entre todos los vecinos de $\mathbf{P}_i$ de los patrones de salida.

Entonces, cualquier

$$q \geq \max\left(1, \frac{\varepsilon_{oi}^{A}}{2p} + 1, \frac{\varepsilon_{oi}^{B}}{2n} + 1\right) \quad \text{(B.18)}$$

puede ser elegido para asegurar que ($A_i$,$B_i$) será recuperado.

## *Simpson (1990)*

En este trabajo, Simpson [33] de la División de Electrónica en San Diego California examina y compara la capacidad y eficiencia de almacenamiento de las memorias autoasociativas, y bidireccionales comunes e intraconectadas de primer y alto orden.

Las memorias autoasociativas de primer orden, comúnmente referidas como memorias Hopfield (HAM), se visualizan como una red neuronal de una capa. En este tipo de memorias se realiza la correlación entre dos elementos (o neuronas) de dicha capa. La topología de su conexión se muestra en la figura B.5.

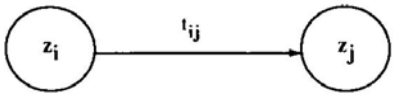



**Figura B.5** Topología de la conexión de los elementos de una memoria autoasociativa de primer orden

En la figura B.6 se pueden observar las conexiones de una memoria autoasociativa de segundo orden. En este tipo de memoria, la correlación se realiza entre dos elementos de la capa y todos los restantes.

La memoria bidireccional de primer orden, descrita ya en la primera parte de esta sección, y propuesta por Kosko, se visualiza como una red neuronal de dos capas en donde las conexiones entre sus elementos se muestran en la figura B.7.

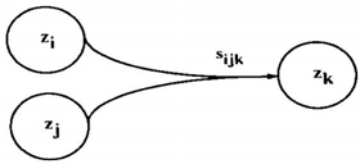



**Figura B.6** Topología de la conexión de los elementos de una memoria autoasociativa de segundo orden.

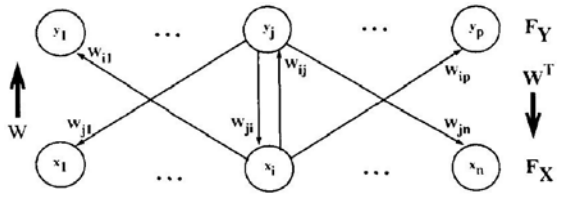



**Figura B.7** La topología de la BAM se visualiza como una red neuronal de dos capas. Las conexiones de primer orden almacenan la correlación entre el elemento de una capa ($F_X$) con un elemento de la segunda capa ($F_Y$).

En la figura B.8 se muestran las conexiones en una BAM de segundo orden. La BAM de segundo orden se visualiza, al igual que la de primer orden, como una red neuronal de dos capas. En esta memoria, las correlaciones entre un par de elementos de una capa ($F_X$) con un elemento de la otra capa ($F_Y$) se guarda en la matriz **U** que tiene dimensiones de *n* x *n* x *p*, siendo *n* la dimensión de los vectores de entrada y *p* la dimensión de los patrones de salida. En la matriz **V** se guardan las correlación de un par de elementos de la capa $F_Y$ con un elemento de la capa $F_X$, esta matriz tiene dimensiones de *p* x *p* x *n*.

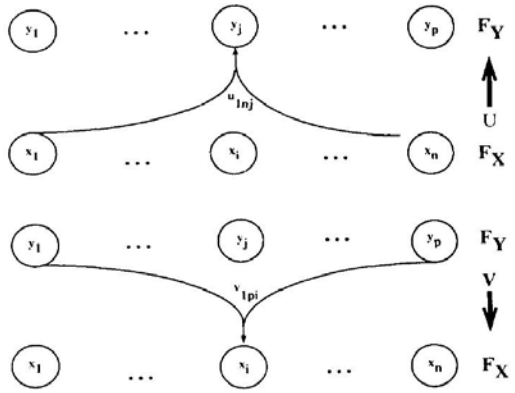



**Figura B.8** Topología de la conexión de una BAM de segundo orden.

Un efecto no deseado en la codificación de la BAM es que el complemento de los patrones almacenados, también se codifica. Por ejemplo, se asume que el par de patrones ($X_1$, $Y_1$), $X_1$ = (1 –1 1) y $Y_1$ = (1 –1 –1 –1) se codifica en una matriz de 3 x 4 que representa a la BAM.

Por defecto, los pares de patrones $(X_1^c, Y_1^c)$, $X_1^c = (-11-1)$ y $Y_1^c = (1111)$, también serán codificados. Si un nuevo par de patrones ($X_2$, $Y_2$), $X_2$ = (1 –1 1) y $Y_2$ = (1 –1 –1 –1) necesitan ser codificados, esto sería imposible con la metodología de codificación de la BAM, debido a que $X_1^c = X_2$.

Una forma de atacar este problema sería creando una memoria autoasociativa a partir de una heteroasociativa, esto es, se concatenan las dos capas $F_X$ y $F_Y$ en una sola que sería $F_Z$, de la siguiente manera:

$$(X \mid Y) = (x_1, x_2, ..., x_n, y_1, y_2, ..., y_p)$$
$$= (z_1, z_2, ..., z_q) = Z$$

donde $q = n + p$. La memoria que utiliza la ecuación anterior se denomina HAM con heterocorrelación y sugiere que al adicionar intraconexiones se puede eliminar el problema del complemento.

La BAM intraconectada (IBAM) se basa en la memoria anterior y adiciona conexiones entre elementos de la misma capa, lo que resulta en el mejoramiento de la capacidad de almacenamiento y también remueve la restricción del complemento. La red resultante se muestra en la figura B.9. Este tipo de memoria se visualiza como una red neuronal de dos capas, las interconexiones (entre elementos de las dos capas $F_X$ y $F_Y$) se almacenan en la matriz C; las intraconexiones (conexiones entre elementos de la misma capa) en la capa $F_X$, se almacenan en la matriz **A**; y las intraconexiones en la capa $F_Y$ se almacenan en la matriz **B**.

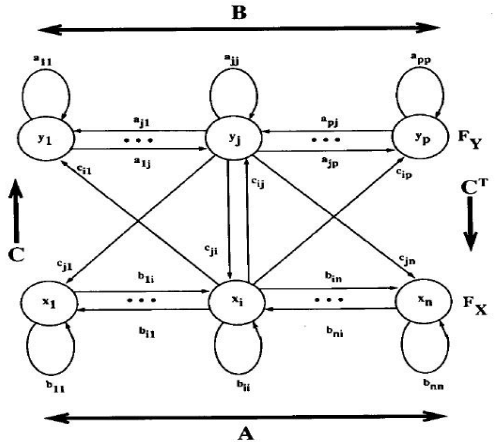



**Figura B.9** Topología de conexión de una BAM intraconectada de primer orden.

Simpson muestra que la capacidad de almacenamiento de las BAM (HHAM, BAM e IBAM) de primer orden se duplica mediante las BAM de segundo orden.

En el caso de la razón de información, que es una medida que describe la eficiencia del almacenamiento, mostró que las BAM de primer orden exhiben una mejor eficiencia en el almacenamiento que las BAM de segundo orden.

## *Wang Y. (1991)*

Wang *et al* [19], del Departamento de Ingeniería Eléctrica y Computación de la Universidad de Califormia, derivan las condiciones necesarias y suficientes para los pesos que conforman la matriz de correlación de una BAM, la cual debe garantizar la recuperación de todos los patrones entrenados, cuando existe la solución. El método de programación lineal/entrenamiento múltiple (*Linear Programming/Multiple Training*, LP/MT) se utiliza para determinar los pesos que satisfacen las condiciones necesarias para que exista una solución factible. El método de entrenamiento múltiple secuencial (*Sequential Multiple Training*, SMT) produce enteros para los pesos, donde estas ponderaciones son multiplicidades de entrenamiento de los patrones entrenados.

Utilizando un conjunto de pares entrenados $\{P_i = (A_i, B_i) | \ i = 1, ..., N\}$, donde

$$A_i = (a_{i1}, a_{i2}, ..., a_{in}),$$
$$B_i = (b_{i1}, b_{i2}, ..., b_{in}),$$

La matriz de correlación original utilizada por Kosko es:

$$M_o = \sum_{i=1}^{N} X_i^T Y_i \tag{B.19}$$

donde

$$X_i = (x_{i1}, x_{i2}, ..., x_{in}),$$
$$Y_i = (y_{i1}, y_{i2}, ..., y_{ip}),$$

Sea $P_i^{'} = \left(A_i^{'}, B_i^{'}\right)$ el vecino de $(A_i, B_i)$ alejado una distancia de Hamming igual a uno, entonces

$$\eta_{ij} = A_i X_j^t Y_j B_i^t - A_i^{'} X_j^t Y_j B_i^{'t} \tag{B.20}$$

donde $\eta_{ij}$ es la diferencia de energías contribuidas por $P_j$ hacia $P_i$' y $P_i$.

Si $\eta_{ij} < 0$ entonces el vecino $P_i$' tiene más energía negativa que $P_i$ desde $P_j$, tal que $P_i$ es relativamente más inestable; si $\eta_{ij} > 0$ entonces $P_i$ tiene más energía negativa que $P_i$' desde $P_j$, tal que $P_i$ es relativamente más estable.

La condición para que el par $P_i = (A_i, B_i)$ sea recuperado utilizando la matriz de correlación con múltiple entrenamiento:

$$M = \sum_{i=1}^{p} q_i X_i^T Y_i \tag{B.21}$$

es:

$$\sum_{j=1}^{p} q_i \eta_{ij} \geq 0 \tag{B.22}$$

La desigualdad anterior puede resolverse mediante métodos de programación lineal, esto si la solución existe. El método utilizado en este trabajo es el método *Simplex*.

En la figura B.10 se pueden observar las comparaciones de las capacidades de almacenamiento entre la BAM original [14], el SMT (*Sequential Multiple Training*) [28] y la BAM implementada con el método LP/MT utilizado en este trabajo.

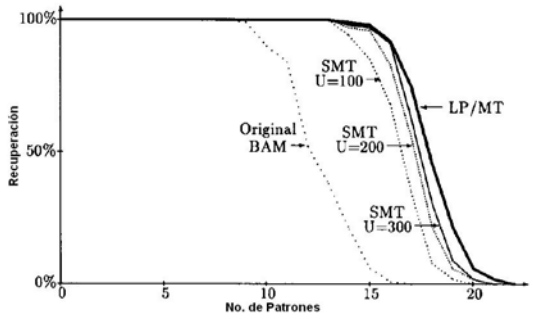



**Figura B.10** Comparación de la capacidad de almacenamiento de tres métodos: la BAM de Kosko, SMT y LP/MT.

## *Leung (1991)*

Kosko [14] probó que la energía de la BAM decrecía durante el proceso iterativo de recuperación, y que al llegar a un mínimo local, la asociación inicial convergía a un punto fijo (o estado estable), el cual podría ser una asociación perteneciente al conjunto fundamental.

Por tanto, Leung *et al* [17] concluyeron que si cada una de las parejas del conjunto fundamental era un estado estable, entonces se podrían recuperar todos los patrones.

En este trabajo, la BAM son dos capas de entrada y salida, una de ellas se denota como el campo $f_A$ y tiene $L_A$ neuronas contenidas en el vector $\vec{A}$. La otra capa, denotada como el campo $f_B$, tiene $L_B$ neuronas contenidas en el vector $\vec{B}$.

Asegurar que todos los pares $(\vec{A}, \vec{B})$ son puntos fijos, es equivalente a encontrar una matriz de interconexión, M*, tal que

$$F_B = \text{sgn}(M^* F_A)$$
$$F_A = \text{sgn}(M^{*T} F_B)$$

Si se utilizan dos matrices de interconexión diferentes, $M_1^*$ (del campo $f_A$ al campo $f_B$) y $M_2^*$ (del campo $f_B$ al campo $f_A$), el requisito para que todos los pares sean puntos fijos se convierte en

$$F_B = \text{sgn}(M_1^* F_A)$$
$$F_A = \text{sgn}(M_2^* F_B)$$

Una condición suficiente para establecer la relación anterior es que

$$F_B = M_1^{*} F_A \quad \text{y} \quad F_A = M_2^{*} F_B$$

Si los vectores $\vec{A}_k$'*s* y $\vec{B}_k$'*s* son linealmente independientes, se pueden definir dos matrices de rotación $\mathbf{R}_A$ y $\mathbf{R}_B$ con dimensiones $L_A$ x $L_A$ y $L_B$ x $L_B$ respectivamente, tales que:

$$Y_A = R_A F_A = \begin{bmatrix} Y_{A,0} \\ \cdots \\ \mathbf{0}_{L_A - N, N} \end{bmatrix} \tag{B.23}$$

$$Y_B = R_B F_B = \begin{bmatrix} Y_{B,0} \\ \cdots \\ \mathbf{0}_{L_B - N, N} \end{bmatrix} \tag{B.24}$$

$Y_{A,0}$ y $Y_{B,0}$ son matrices triangulares superiores de dimensión $N$ x $N$, y $\mathbf{0}_{LA\text{-}N,N}$ y $\mathbf{0}_{LB\text{-}N,N}$ son matrices cero de dimensiones ($L_A$-$N$) x $N$ y ($L_B$-$N$) x $N$ respectivamente. Debido a que $\mathbf{R}_A$ y $\mathbf{R}_B$ son matrices ortonormales,

$$F_A = \hat{R}_A^T Y_A \tag{B.25}$$

y

$$F_B = \hat{R}_B^T Y_B \tag{B.26}$$

Debido a la poca densidad de $Y_A$ y $Y_B$, la matriz de rotación $\mathbf{R}_A$ puede reducirse en dimensión eliminado las $L_A$-$N$ filas después de la ($N$+1)-ésima fila. De igual forma, se pueden eliminar $L_B$-$N$ filas después de la ($N$+1)-ésima fila de la matriz de rotación $\mathbf{R}_B$. Después de la eliminación se tiene,

$$F_A = \hat{R}_A^T Y_{A,0} \tag{B.27}$$

y

$$F_B = \hat{R}_B^T Y_{B,0} \tag{B.27}$$

respectivamente. Sustituyendo se obtiene:

$$\hat{R}_B^T Y_{B,0} = M_1^{*} \hat{R}_A^T Y_{A,0} \tag{B.28}$$

y

$$\hat{R}_A^T Y_{B,0} = M_2^{*} \hat{R}_B^T Y_{B,0} \tag{B.29}$$

Nótese que $\hat{R}_B \hat{R}_B^T$ y $\hat{R}_A \hat{R}_A^T$ son matrices identidad con dimensiones $N$ x $N$, por tanto se obtienen las matrices $M_1^{*}$ y $M_2^{*}$ como sigue:

$$M_1^* = \hat{R}_B^T Y_{B,0} Y_{A,0}^{-1} \hat{R}_A \tag{B.30}$$

$$M_2^* = \hat{R}_A^T Y_{A,0} Y_{B,0}^{-1} \hat{R}_B \tag{B.31}$$

donde $M_1^*$ y $M_2^*$ son la matriz directa (de $f_A$ a $f_B$) y la matriz hacia atrás (de $f_B$ a $f_A$) respectivamente.

Ya que la probabilidad de que $N$ vectores aleatorios bipolares con dimensión $r$ ($N \leq r$) sean linealmente independientes tiende a 1 cuando $r$ es suficientemente grande, este esquema muestra una capacidad del min($L_A$, $L_B$).

## *Srinivasan (1991)*

En este trabajo, Srinivasan y Chia [34] del Departamento de Ingeniería Eléctrica de la Universidad de Singapore, proponen la eliminación de los estados espurios o no deseados, mediante el proceso de *unlearning* o cancelación.

El primer paso del método es crear la matriz $M$ al estilo de Kosko [14]. Después, se le presenta un par $(X_{s1} = X_1, Y_{s1} = Y_1)$ inicial, que pertenece al conjunto fundamental, si esta asociación no recupera el patrón esperado, entonces quiere decir que el par convergió a un estado espurio. El siguiente paso es eliminar dicho estado, modificando la matriz $M$ de la siguiente forma:

$$M_{u1} = 2M - X_{s1}^T Y_{s1} \tag{B.32}$$

Se puede observar que se utiliza un entrenamiento múltiple con un factor de 2, combinado con un término negativo para cancelar el estado espurio.

Si, de nuevo, se encuentra otro estado espurio entonces se modifica la matriz **M**, como sigue:

$$M_{u2} = 3M - X_{s1}^T Y_{s1} - X_{s2}^T Y_{s2} \tag{B.33}$$

En general, la estrategia de codificación para almacenar $N$ pares de vectores en presencia de $S$ estados espurios, mediante el *unlearning*, es:

$$M_{us} = \sum_{k=1}^{N} q_k X_k^T Y_k - \sum_{s=1}^{S} q_{ss} X_{ss}^T Y_{ss} \tag{B.34}$$

donde ($X_k$, $Y_k$) son los patrones que van a almacenarse y ($X_{ss}$, $Y_{ss}$) son los estados espurios. $q_k$ y $q_{ss}$ son los factores del entrenamiento múltiple y eliminación, respectivamente. Sus valores deben ser determinados experimentalmente.

En la figura B.11 se muestran las capacidades de almacenamiento para los métodos de Kosko, entrenamiento múltiple (MT) [28] y para la estrategia que propuesta en este trabajo,

el unlearning. De la figura se puede observar que este último método es mejor que los otros dos.

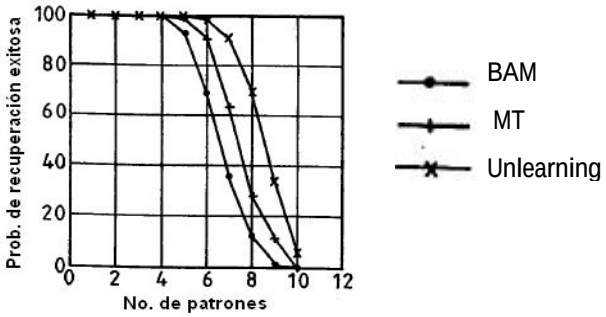



**Figura B.11** Capacidad de almacenamiento para la BAM de Kosko, Entrenamiento Múltiple (MT) y Unlearning (U). Para $n = p = 40$.

## *Wang W. (1992)*

Wang *et al* [35], de la Universidad Nacional Central de Taiwán, proponen una modificación a la BAM exponencial (EBAM) [30] al agregar un término al exponente de autocorrelación de manera que se mejore la capacidad de la memoria. El término agregado introduce una intraconexión en la EBAM que permite remover los problemas adicionales de codificación y la suposición de continuidad.

En la EBAM se asume que existen *m* pares almacenados $\{(X^i, Y^i)\}_{i=1}^m$ donde $X^i \in \{-1,1\}^n$ y $Y_i \in \{-1,1\}^p$.

Las reglas de actualización de la EBAM se muestran a continuación:

$$Y = f(X) = \text{sgn}\left\{\sum_i^m Y^i \alpha^{\langle X^i, X\rangle}\right\} \tag{B.35}$$

$$X' = g(Y) = \text{sgn}\left\{\sum_{i=1}^m X^i \alpha^{\langle Y^i, Y\rangle}\right\} \tag{B.36}$$

donde <**A**, **B**> el producto interno de los vectores **A** y **B**, $\alpha$ es un número mayor que la unidad, y sgn ($x$) = 1 si $x \geq 0$ y sgn($x$) = -1 de otra forma.

La MEBAM tiene el mismo conjunto fundamental.

A continuación se muestran las reglas de actualización de la MEBAM:

$$Y' = f(X,Y) = \text{sgn}\left\{\sum_i^m Y^i \left[\alpha^{\langle X^i, X\rangle} + \alpha^{\langle Y^i, Y\rangle}\right]\right\} \tag{B.37}$$

$$X' = f(X, Y') = \text{sgn}\left\{\sum_i^m X^i \left[\alpha^{\langle X^i, X\rangle} + \alpha^{\langle Y^i, Y'\rangle}\right]\right\} \tag{B.38}$$

La capacidad de almacenamiento se mide por el número de recuperaciones exitosas de *m* pares generados aleatoriamente. La dimensión de los patrones fue de $n = p = 16$ y $\alpha = e$. La generación desigual de probabilidades de 1 y –1 en $Y^i$ son para propósito de analizar el comportamiento de ambas BAM respecto a la discontinuidad.

La capacidad de la MEBAM se comparó con la de la EBAM y los resultados se muestran en la figura B.12.

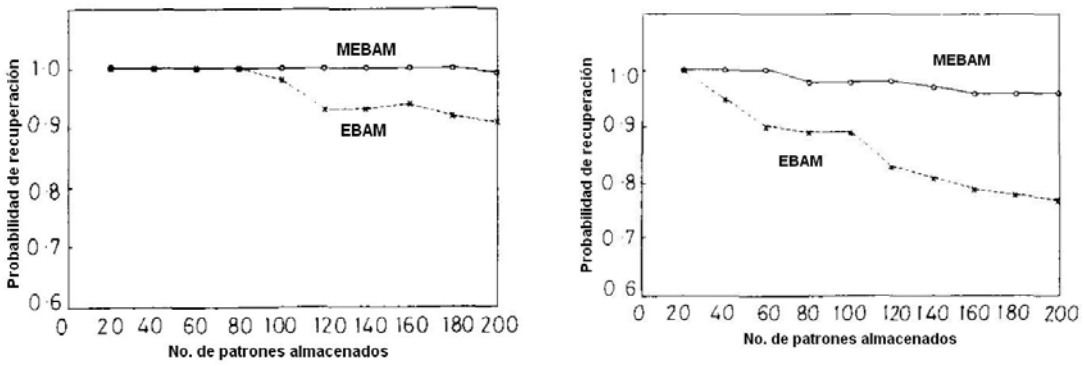



**Figura B.12** Gráfica de comparación de las capacidades de almacenamiento entre la MEBAM y la EBAM. (a) P(1) = 0.75, P(-1) = 0.25 para $\{Y^i\}$; (b) P(1) = 0.80, P(-1) = 0.20 para $\{Y^i\}$

De las gráficas se puede observar que entre mayor es la diferencia entre la generación de probabilidades de 1 y –1, es menor la posibilidad de que se cumpla la condición de suposición de continuidad propuesta por Kosko:

$$\frac{1}{n} H(X^i, X^j) \approx \frac{1}{p} H(Y^i, Y^j)$$

Sin embargo, también se observa que la MEBAM tiene un mejor comportamiento con respecto a la EBAM en cuanto a la continuidad de los patrones.

## *Jeng (1992)*

La BAM de alto orden (HOBAM) [29] y la BAM exponencial (EBAM) [30] se propusieron para mejorar la capacidad de almacenamiento. Desafortunadamente, estos esquemas basados en sólo en la modificación de las reglas de actualización no garantizan sus propiedades de estabilidad. Para lograr un mejor rendimiento y una mejor capacidad de

almacenamiento Jeng *et al* [36], del Departamento de Ingeniería Eléctrica de la Universidad Nacional de Taiwán, propusieron un modelo general de correlación para la BAM a través una condición suficiente que permite la estabilidad de los modelos de memoria HOBAM y EBAM. En este modelo, el estado de cada neurona se determina no sólo por la activación de neuronas de la otra capa, sino que también depende de activaciones previas. Este modelo se denomina memoria asociativa bidireccional estable de alto orden (*Stable High order BAM*, SHBAM).

Este modelo memoriza *m* pares de asociaciones bipolares $(X_k,Y_k)$, $k = 1, ..., m$, $X^i \in \{-1,1\}^n$ y $Y_i \in \{-1,1\}^p$.

Las actualizaciones del modelo general son:

$$X' = \operatorname{sgn}\left\{\sum_{k=1}^{m} X_k g_k\left(\langle X, X_k\rangle, \langle Y, Y_k\rangle\right)\right\} \tag{B.39}$$

$$Y' = \operatorname{sgn}\left\{\sum_{k=1}^{m} Y_k f_k\left(\langle X', X_k\rangle, \langle Y, Y_k\rangle\right)\right\} \tag{B.40}$$

donde *X'* y *Y'* son los patrones del siguiente estado de *X* y *Y*, respectivamente, $\langle X, Y\rangle$ denota el producto interno de *X* y *Y*, y $f_k()$ y $g_h()$ son funciones de ponderación.

La condición suficiente para que el modelo general de la BAM sea bidireccionalmente estable se describe a continuación.

La memoria asociativa bidireccional descrita por las ecuaciones anteriores es bidireccionalmente estable si $f_k(x, y)$ son funciones monotónicamente no decrecientes de *y* y $g_k(x, y)$ son funciones monótonamente no decrecientes de *x* cuando operan en la región $[-n, n] \times [-m, m]$, y si $f_k(x, y)$ y $g_k(x, y)$ satisfacen las siguientes dos ecuaciones:

$$f_k(x, y) = \frac{\partial e_k(x, y)}{\partial y} \tag{B.41}$$

$$g_k(x, y) = \frac{\partial e_k(x, y)}{\partial x} \tag{B.42}$$

donde $\{e_k(x, y), k = 1, ..., p\}$ es un conjunto limitado de funciones diferenciables.

A partir de esta condición, las ecuaciones de actualización se pueden definir como:

$$X' = \operatorname{sgn}\left\{\sum_{k=1}^{m} X_k \langle X, X_k\rangle^{r-1}, \langle Y, Y_k\rangle^{s}\right\} \tag{B.43}$$

$$Y' = \text{sgn}\left\{\sum_{k=1}^{m} Y_k \langle X', X_k \rangle^r, \langle Y, Y_k \rangle^{s-1}\right\} \tag{B.44}$$

donde *r* y *s* son enteros positivos.

La SHBAM de tercer orden se puede formar eligiendo $r = s = 2$. Comparándola con la HOBAM de tercer orden la SHBAM tiene una mejor capacidad de almacenamiento.

Se realizaron simulaciones para examinar las capacidades de la SHBAM y la HOBAM. En las pruebas, cada uno de los patrones pertenecientes al conjunto fundamental se le presentaron a las memorias como patrones de entrada. Las probabilidades de recuperación y el porcentaje de recuperaciones exitosas se calcularon tomando el promedio de 100 pruebas independientes con varias asociaciones generadas aleatoriamente. La figura B.13 muestra las probabilidades de recuperación de la SHABM y de la HOBAM de tercer orden.

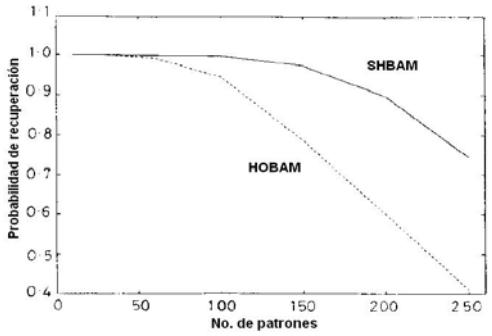



**Figura B.13** Gráfica de comparación de las capacidades de almacenamiento entre la HOBAM y SHBAM, ambas de tercer orden.

## *Yu (1992)*

Cuando los patrones son ortogonales entre sí, es ideal recuperarlos cuando se le presentan a una matriz de correlación. Para un conjunto de patrones entrenados que no son ortogonales es necesario una función de umbral no lineal que actúe como un filtro cuando el efecto del ruido no sea tan evidente en el proceso de asociación, lo que significa que el entrenamiento del conjunto de patrones debe tener el requerimiento de continuidad en el sentido de la distancia de Hamming. Pero esta restricción es tan estricta que es generalmente imposible cumplirla en la práctica.

La principal diferencia entre la BAM de Kosko [14] y la BAM generalizada (*Generalized Bidirectional Associative Memory*, GBAM) [37] radica en el hecho de que la GBAM permite asignar diferentes factores de peso a cada uno de los bits de los vectores de los patrones durante la etapa de aprendizaje.

Yu *et al*, del Instituto de Ingeniería de Sistemas de la Universidad de Tecnología Dalian en China, afirman que: “desafortunadamente, no existe un método efectivo de asegurar que

cada uno de los pares de patrones sea un mínimo local de la función de energía *E*". Lo que parece desacreditar a los métodos que pretenden que cada uno de los patrones del conjunto fundamental corresponda a un mínimo local de *E*.

Para que la BAM de Kosko tenga recuperación perfecta, es necesario que se cumpla la condición de continuidad en el sentido de la distancia de Hamming, esta condición se define a continuación.

Considérense *m* pares de patrones almacenados, llamados el conjunto de patrones entrenados

$$(A_1,B_1),\ (A_2,B_2),\ ...,\ (A_m,B_m)$$

donde

$$A_i = \left(a_1^{(i)}, a_2^{(i)}, ..., a_n^{(i)}\right) \in (0,1)^n$$
$$B_i = \left(b_1^{(i)}, b_2^{(i)}, ..., b_p^{(i)}\right) \in (0,1)^p$$

La condición es

$$\frac{1}{n} H(A_i, A_j) = \frac{1}{p} H(B_i, B_j) \qquad \forall\ i \in \{1, 2, ..., n\} \text{ y } j \in \{1, 2, ..., p\}$$

Donde H(,) es la distancia de Hamming entre dos vectores.

Como ya se había mencionado, este criterio es muy difícil de cumplir, por lo que en este trabajo se propone asignar factores positivos a cada bit de los vectores de patrones, por lo que se define un vector de peso $\alpha$ para los vectores de entrada y $\beta$ para los vectores de salida.

$$\alpha = \{\alpha_1,\ \alpha_2,\ ...,\ \alpha_n\}$$

$$\beta = \{\beta_1,\ \beta_2,\ ...,\ \beta_p\}$$

Los patrones ponderados se definen como:

$$\tilde{X} = \alpha \bullet X = \{\alpha_1 x_1, \alpha_2 x_2, \ldots, \alpha_n x_n\} \tag{B.45}$$

$$\tilde{Y} = \beta \bullet Y = \{\beta_1 y_1, \beta_2 y_2, \ldots, \beta_p y_p\} \tag{B.46}$$

Basándose en el entrenamiento con los nuevos patrones ponderados, se puede redefinir la matriz **M**:

$$\tilde{M} = \sum_{i=1}^{m} \tilde{X}_i^T \tilde{Y}_i$$

A partir de este cambio, entonces el criterio de continuidad se convierte en:

$$\frac{1}{\sum_{i=1}^{n}\alpha_i^2} H\alpha\left(A_i, A_j\right) = \frac{1}{p} H\left(B_i, B_j\right) \tag{B.47}$$

$$\frac{1}{\sum_{i=1}^{p}\beta_i^2} H\beta\left(B_i, B_j\right) = \frac{1}{n} H\left(A_i, A_j\right) \tag{B.48}$$

Las dos ecuaciones anteriores son la condición de continuidad en el sentido de la distancia ponderada de Hamming. Existe una mayor posibilidad de que se cumpla el criterio de la GBAM que el de la BAM, debido a que $\alpha$ y $\beta$ se pueden elegir libremente.

Aunque una elección apropiada de $\alpha$ y $\beta$ permite un mejor rendimiento de la GBAM, no existe un método para su elección por lo que se deben obtener de forma experimental, lo que es una clara desventaja.

## *Lee (1993)*

Lee y Wang [38] en la Universidad Nacional Central de Taiwán, utilizan la técnica llamada *Correlation Significance*, en donde se introducen las ponderaciones entre las asociaciones aprendidas y entre las neuronas de las diferentes capas ($F_x$ y $F_y$), de tal manera que se pueda mejorar la capacidad de almacenamiento de la BAM de Kosko [14]. El camino para lograr esto es construir una función de error, basándose en el método del gradiente descendiente, de forma tal que se puedan recuperar el máximo de asociaciones entrenadas.

Otros trabajos, como el de Haines *et al* [16] cambian los umbrales para incrementar la capacidad de su memoria, Wang *et al* [19, 31], con el mismo objetivo, introducen el concepto de entrenamiento múltiple agregándole pesos a cada termino de correlación.

El método *Correlation Significance* (CS) para memorias asociativas bidireccionales se describe a continuación.

Sean $\{(X^s, Y^s)\}_{s=1}^{N}$ los $N$ conjuntos de asociaciones almacenados, donde $X^s \in \{-1,1\}^n$ y $Y^s \in \{-1,1\}^p$.

La regla de actualización para la fase de recuperación en la BAM de Kosko es:

$$Y = \text{sgn}\{XM\} \tag{B.49}$$

$$X' = \text{sgn}\{YM^T\} \tag{B.50}$$

donde $M = \sum_{s=1}^{m} X^{sT} Y^s$ y sgn($\alpha$) = 1 para $\alpha \geq 0$, sgn($\alpha$) = -1 de otra forma. Para incrementar la capacidad de la BAM, los pesos de conexión se construyen como sigue:

$$m_{ij} = \sum_{s=1}^{N} x_i^s \omega_{ij} y_j^s \tag{B.51}$$

donde $m_{ij}$ denota los pesos de conexión entre la *i*-ésima neurona de la capa $F_x$ y la *j*-ésima neurona de la capa $F_y$, y $x_i^s$ y $y_j^s$ son la *i*-ésima y *j*-ésima componente de los vectores $\mathbf{x}^s$ y $\mathbf{y}^s$, respectivamente. $\omega_{ij}$ es el parámetro de importancia correspondiente a $x_i^s$ y $y_j^s$. Nótese que si $\omega_{ij}$ es despreciada, entonces los pesos de conexión son idénticos a la matriz *M* de Kosko. También se puede observar que si $\omega_{ij} = \omega^s$, entonces se está utilizando el método de entrenamiento múltiple de Wang *et al* [19], con la única diferencia que en este último trabajo se utiliza la técnica de programación lineal para encontrar el valor de $\omega^s$.

En el CS se utiliza el método del gradiente descendiente para encontrar $\omega_{ij}$, sólo que también se actualiza $\omega_{ij}$ en cada iteración de la etapa de recuperación:

$$\omega_{ij}^s(k+1) = \omega_{ij}^s(k) - \eta g_{ij}^s$$

donde $g_{ij}^s$ es el gradiente de la función de error con respecto a $\omega_{ij}^s$ y $\eta$ es el tamaño de paso ($0 < \eta < 1$).

En la figura B.14 se muestra la comparación de las capacidades de almacenamiento entre la BAM de Kosko, la de Wang y la CS. En las simulaciones se tomaron los valores de η = 0.1, p se incrementó desde 1 hasta 10 patrones, las dimensiones de los vectores de entrada y salida se fijaron a $n = p = 34$. Las *p* asociaciones entrenadas se eligieron de forma arbitraria de entre un conjunto de 100 elementos generados aleatoriamente.

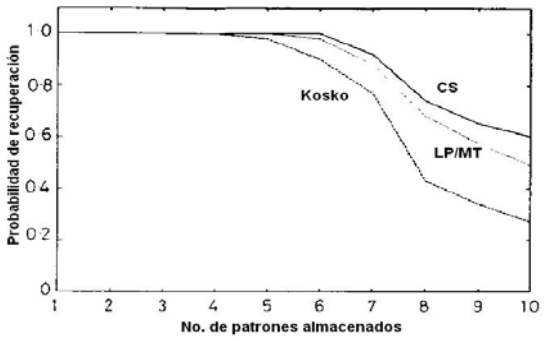



**Figura B.14** Gráfica de comparación de la capacidad de almacenamiento entre el CS, LP/MT y el método de Kosko.

Se puede observar de la figura que la probabilidad de recuperación desciende conforme el número de patrones aprendidos aumenta, sin embargo el rendimiento de la BAM con CS es mejor que el rendimiento de las otras BAM.

## *Perfetti (1993)*

Perfetti [39], de la Universidad de Perugia en Italia, presenta un algoritmo de aprendizaje para memorias asociativas bidireccionales en donde la estrategia utilizada se formula como un problema de optimización convexo, el cual se resuelve mediante el método del gradiente descendiente. Este algoritmo pretende lograr que el máximo de los patrones aprendidos sean estados estables. Si se logra lo anterior, entonces es posible incrementar la capacidad de almacenamiento.

El algoritmo de aprendizaje se describe a continuación.

Sea $\{X^{(k)}, Y^{(k)}, k = 1, ..., N\}$ que denota el conjunto de asociaciones que serán almacenadas; esto es, $X^{(k)} \in \{-1,1\}^n$ y $Y^{(k)} \in \{-1,1\}^p$.

De acuerdo a la regla utilizada en el proceso de recuperación de la BAM de Kosko [14], se puede obtener de forma directa la demostración de que, una asociación $(x^{(k)},y^{(k)})$ representa un estado estable de la BAM si y sólo si:

$$x_i^{(k)} = \text{sgn}\left( \sum_{j=1}^{p} m_{ij} y_j^{(k)} \right) \quad \text{para } i = 1, ..., n$$

$$y_j^{(k)} = \text{sgn}\left( \sum_{i=1}^{n} m_{ij} x_i^{(k)} \right) \quad \text{para } j = 1, ..., p$$

donde $\text{sgn}(\alpha) = 1$ para $\alpha \geq 0$, $\text{sgn}(\alpha) = -1$ de otra forma. $m_{ij}$ es el $ij$-ésimo elemento de la matriz $\boldsymbol{M}$; las dos ecuaciones anteriores pueden rescribirse como:

$$\sum_{j=1}^{p} m_{ij} x_i^{(k)} y_j^{(k)} > 0 \quad \text{para } i = 1, ..., n \tag{B.52}$$

$$\sum_{i=1}^{n} m_{ij} x_i^{(k)} y_j^{(k)} > 0 \quad \text{para } j = 1, ..., p \tag{B.53}$$

La matriz $\boldsymbol{M}$ se debe determinar de manera que se satisfagan las restricciones de diseño de $n$, $p$ y $N$. Sea $\gamma \in \Re^{np}$ que denota el vector cuyos elementos son los pesos de conexión $m_{ij}$. El espacio de solución de las ecuaciones anteriores es un poliedro convexo $\Omega$. Un punto $\gamma \in \Omega$ se llama punto factible.

Para hallar la solución de las desigualdades, se puede utiliza el método del gradiente descendiente para reducir el esfuerzo computacional.

Considere la función de costo:

$$\psi(\gamma) = \frac{1}{2}\sum_{k=1}^{N}\left[\sum_{i=1}^{n}\left(F_x^-(k,i)\right)^2 + \sum_{j=1}^{p}\left(F_y^-(k,j)\right)^2\right] \tag{B.54}$$

donde

$$F_x(k,i) = \sum_{j=1}^{p} m_{ij} x_i^{(k)} y_j^{(k)} - \delta \tag{B.55}$$

$$F_y(k,j) = \sum_{i=1}^{n} m_{ij} x_i^{(k)} y_j^{(k)} - \delta \tag{B.56}$$

y $F_x^- = \min\{0, F_x\}$, $F_y^- = \min\{0, F_y\}$. $\delta$ es una constante real positiva que es un parámetro que permite que los *basins* de atracción, de los pares almacenados, sean maximizados. Un incremento de $\delta$ corresponde a un incremento en la robustez del comportamiento de la BAM, cuando los pesos de conexión se implementan con precisión.

La minimización de $\psi(\gamma)$ mediante el método del gradiente descendiente garantiza la convergencia a un punto estacionario, esto es, garantiza una matriz **M** factible:

$$m_{ij}(t+1) = \sigma\left[m_{ij}(t) - \eta\sum_{k=1}^{N}\left(F_x^-(k,i) + F_y^-(k,j)\right)x_i^{(k)} y_i^{(k)}\right] \tag{B.57}$$

donde $0 < \eta < 1$, y $\sigma(x)$ es una función de límite rígido: $\sigma(x) = x$, si $-1 \le x \le 1$; $\sigma(x) = 1$ si $x > 1$; $\sigma(x) = -1$ si $x < -1$.

En la figura B.15 se muestra la comparación de la BAM de Kosko, BAM-MT [18, 34] y la BAM de este trabajo. Los valores de los parámetros usados son: $\delta = 0.1$, $\eta = 0.01$, el rango del número de patrones almacenados, $N$, fue desde 1 hasta 12, con $n = p = 34$. Para cada $N$, se probaron 100 conjuntos independientes. Cada conjunto consistía de $N$ asociaciones generadas aleatoriamente. Si todos los pares almacenados eran estados estables la prueba se consideraba exitosa.


**Figura B.15** Gráfica de comparación de las capacidades de almacenamiento de Kosko, entrenamiento múltiple y el método presentado en este trabajo.

## *Leung (1993)*

La BAM de Kosko [14] está basada en el método de correlación cuya capacidad es muy pequeña aún cuando el número de patrones también lo es. Para incrementar la capacidad, Wang *et al* [31] introdujeron dos esquemas de codificación: el aumento *dummy* y el múltiple entrenamiento. El aumento *dummy* incrementa el rendimiento de la recuperación con el costo de agregar muchas neuronas, mientras la capacidad de la BAM con el esquema del entrenamiento múltiple es aún baja. Leung y Cheung [17] introdujeron el algoritmo de codificación Householder (*Householder Coding Algorithm*, HCA) en donde la capacidad de al BAM era el valor mínimo de las dimensiones de los vectores de entrada y salida. Sin embargo, este algoritmo no converge a un estado estable cuando el vector inicial no pertenece al conjunto fundamental debido a que existen dos matrices de interconexión. El método de codificación mostrado por Leung [40], de la Universidad China de Hong Kong es, en cierta medida, una continuación de [17]; en este trabajo (*Enhanced Householder Coding Algorithm*, EHCA) se reducen las dos matrices de interconexión del HCA a una sola mediante la proyección sobre conjuntos convexos (*Projection On Convex Sets*, POCS).

El EHCA se describe a continuación.

En este trabajo, la BAM son dos capas de entrada y salida, una de ellas se denota como el campo $f_A$ y tiene $L_A$ neuronas contenidas en el vector $\vec{A}$. La otra capa, denotada como el campo $f_B$, tiene $L_B$ neuronas contenidas en el vector $\vec{B}$.

Si es posible encontrar una matriz de interconexión tal que,

$$\boldsymbol{F}_B = \text{sgn}(M^*\mathbf{F}_A) \text{ y } \mathbf{F}_A = \text{sgn}(\mathbf{M}^{*T}\,\mathbf{F}_B) \tag{B.58}$$

entonces cada patrón fundamental $(\vec{A}_k, \vec{B}_k)$ es un punto fijo.

Primero se generan dos matrices de interconexión $M_f$ y $M_b$ para la recuperación, entonces el requerimiento de que cada asociación debe ser un punto fijo se convierte en:

$$\mathbf{F}_B = \text{sgn}(\mathbf{M}_f\mathbf{F}_x) \text{ y } \mathbf{F}_A = \text{sgn}(\mathbf{M}_b{}^T\,\mathbf{F}_B) \tag{B.59}$$

donde $\mathbf{M}_f$ y $\mathbf{M}_b$ son las matrices de interconexión directa y hacia atrás, respectivamente. Si todos los  y todos los  son linealmente independientes, entonces $\mathbf{M}_f$ y $\mathbf{M}_b$ se pueden encontrar utilizando HCA:

$$\mathbf{M}_f = \mathbf{F}_B\mathbf{F}_A^+ \quad \text{y} \quad \mathbf{M}_b = \mathbf{F}_A\mathbf{F}_B^+$$

donde $\mathbf{F}_A^+$ y $\mathbf{F}_B^+$ son las inversas de $\mathbf{F}_A$ y $\mathbf{F}_B$, respectivamente.

$\mathbf{M}_f$ y $\mathbf{M}_b$ definidas en la ecuación anterior, sólo son un caso particular. La solución general, $\mathbf{M}_{f\alpha}$ y $\mathbf{M}_{b\beta}$, es

$$\mathbf{M}_{f\alpha} = \mathbf{F}_{B\alpha}\mathbf{F}_A^+ \quad \text{y} \quad \mathbf{M}_{b\beta} = \mathbf{F}_{A\beta}\mathbf{F}_B^+$$

Si existen $\alpha_{ij} > 0$ y $\beta_{ln} > 0$ donde $\alpha_{ij}$ es el *ij*-ésimo elemento de $\mathbf{F}_A$ y $\beta_{ln}$ es el *ln*-ésimo elemento de $\mathbf{F}_B$, tal que

$$\mathbf{M}_{f\alpha} = \mathbf{M}_{b\beta}^T$$

entonces $\mathbf{M}^* = \mathbf{M}_{f\alpha} = \mathbf{M}_{b\beta}^T$.

Se realizaron simulaciones para comparar la capacidad de la BAM de Kosko, la BAM con HCA y la BAM con EHCA. $n = p = 20$ y 128. Se varió el número de patrones *N*. Para cada vector de entrada y salida la probabilidad de ser 1 y –1 es igual. Cada conjunto fundamental se llama un evento. Si cada patrón fundamental es un punto fijo entonces se tiene un evento exitoso, de otra forma se llama evento fallido.

Las pruebas indicaron que el HCA tiene una ligera mejoría en la capacidad de almacenamiento que la BAM con EHCA, sin embargo, la segunda permite que la asociación inicial sea una versión ruidosa de alguno de los patrones del conjunto fundamental.

La capacidad de la BAM no es una medida significativa si no se considera, también, la capacidad de la corrección de errores. En la figura B.16 se muestra la gráfica de comparación de la capacidad de corrección de errores entre la BAM de Kosko y la BAM con EHCA para $p = n = 32$.

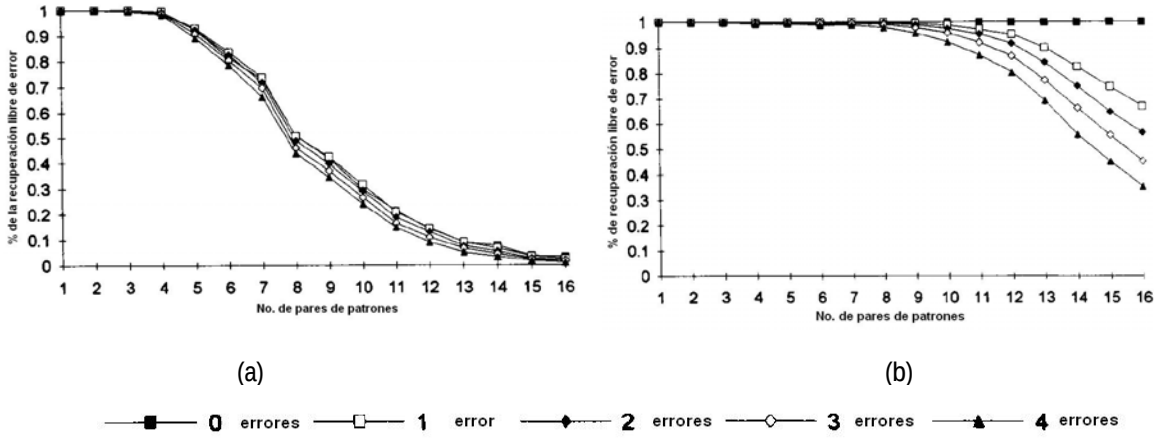



**Figura B.16** Comparación de la capacidad de corrección de errores entre la BAM de: (a) Kosko y (b) BAM con EHCA.

## *Leung (1993)*

El esquema de codificación de Kosko [14] tiene una capacidad muy pequeña. Para incrementar la capacidad de la BAM Leung [17] introdujo el esquema de codificación Householder (*Householder Coding Algorithm*, HCA) donde la capacidad de la BAM era el valor mínimo de entre las dimensiones de los patrones fundamentales. Este algoritmo no converge a un estado estable cuando el vector inicial no pertenece al conjunto fundamental debido a que existen dos matrices de interconexión. Dos años después se presentó el HCA mejorado (*Enhanced Householder Coding Algorithm*, EHCA) [40]. La desventaja de este último es que su regla de aprendizaje es fuera de línea y tiene que trabajar en modo *batch*.

Tomando como idea básica al perceptrón, Leung también propone un aprendizaje bidireccional (*Bidirectional Learning*, BL). En donde encuentra una matriz de interconexión M' que es equivalente a forzar a que cada patrón fundamental sea un punto fijo; esta matriz es tal que

$$\text{sgn}(M'\vec{A}_k) = \vec{B}_k \quad \text{y} \quad \text{sgn}(M'^T\vec{B}_k) = \vec{A}_k$$

donde $k = 1, 2, ..., N$. Las ecuaciones anteriores son simplemente funciones de decisión de un *perceptron* de una capa en sentido bidireccional.

Debido a que BL se basa en la regla del *perceptron*, entonces no puede localizar una buena superficie de decisión. Por tanto, la BAM con BL es, por naturaleza, débil en la corrección de errores. Leung [41], de la Universidad China de Hong Kong, agregó la regla Adaptativa Ho-Kashyap (AHK) al BL (AHKBL), lo que permite generar una superficie de decisión robusta en un *perceptron* de una capa de tal manera que se puede mejorar la habilidad de corregir errores.

El algoritmo de describe a continuación.

Considérese que existe un conjunto de patrones de entrada entrenados $\vec{X}_k \in \{-1,1\}^{Lx}$, donde $k = 1, ..., N$. Cada vector entrenado se asocia con un vector de salida bipolar $\vec{Y}_k \in \{-1,1\}^{Ly}$. En AHK, un vector margen positivo valuado $\vec{D}_k$ es introducido con cada entreda salida ($\vec{X}_k$ y $\vec{Y}_k$). Entrenar el *perceptron* para implementar el mapeo anterior es equivalente a encontrar a $W$ tal que:

$$\vec{Y}_k \otimes \left(W\vec{X}_k\right) = \vec{D}_k > \vec{0} \quad \text{para } k = 1, ..., N$$

donde $\otimes$ es el operador de multiplicación tal que $\mathbf{C} = \mathbf{A} \otimes \mathbf{B}$ significa que $c_i = a_i b_i \; \forall i$, y $\mathbf{A} > \mathbf{B}$ significa que $a_i > b_i \; \forall i$.

En las siguientes ecuaciones, $\mathbf{D}_{f,k}$ se utiliza en el sentido directo con dimensión $L_B$ y $\mathbf{D}_{b,k}$ se usa en el sentido inverso y tiene dimensión $L_A$.

Supóngase que el par inicial presentado a la BAM es $(\vec{X}_k, \vec{Y}_k)$ la corrección de la matriz de interconexión es la siguiente:

- En el sentido directo:

$$\vec{D}_{f,k}^{new} = \vec{D}_{f,k}^{old} + \frac{\rho_1}{2}\left(\left|\varepsilon_{f,k}^{old}\right| + \varepsilon_{f,k}^{old}\right) \tag{B.60}$$

con $$\varepsilon_{f,k}^{old} = \vec{B}_k \otimes \left(\mathbf{M}^{old}\vec{A}_k\right) - \vec{D}_{f,k}^{old}$$

$$\Delta\mathbf{M}_f = \frac{\rho_1\rho_2}{2}\left[\vec{B}_k \otimes \left|\varepsilon_{f,k}^{old}\right| + \left(1 - \frac{2}{\rho_1}\right)\vec{B}_k \otimes \varepsilon_{f,k}^{old}\right]\vec{A}_k^T \tag{B.61}$$

- De acuerdo a esta corrección directa, se actualiza la matriz de interconexión para obtener una matriz temporal $\mathbf{M}^{temp}$

$$\mathbf{M}^{temp} = \mathbf{M}^{old} + \Delta\mathbf{M}_f \tag{B.62}$$

- Ahora, en el sentido inverso se calcula la matriz de interconexión, utilizando la matriz $\mathbf{M}^{temp}$

$$\mathbf{D}_{b,k}^{new} = \mathbf{D}_{b,k}^{old} + \frac{\rho_1}{2}\left(\left|\varepsilon_{b,k}^{old}\right| + \varepsilon_{b,k}^{old}\right) \tag{B.63}$$

con $$\varepsilon_{b,k}^{old} = \vec{A}_k \otimes \left(\mathbf{M}^{temp}\vec{B}_k\right) - \mathbf{D}_{b,k}^{old}$$

$$\Delta\mathbf{M}_b = \frac{\rho_1\rho_2}{2}\vec{B}_k\left[\vec{A}_k \otimes \left|\varepsilon_{b,k}^{old}\right| + \left(1 - \frac{2}{\rho_1}\right)\vec{A}_k \otimes \varepsilon_{b,k}^{old}\right]^T \tag{B.64}$$

- Una vez más se actualiza la matriz de conexión de acuerdo a la corrección en el sentido inverso de manera que se obtiene una nueva matriz de conexión $\mathbf{M}^{new}$

$$\mathbf{M}^{new} = \mathbf{M}^{temp} + \Delta\mathbf{M}_B \tag{B.65}$$

Cabe hacer notar que sólo existe una matriz de conexión.

La condición suficiente para la convergencia del AHKBL es:

$$0 < \rho_1 < 2 \quad \text{y} \quad 0 < \rho_2 < \frac{2}{\max(L_A, L_B)}$$

Se realizaron simulaciones para comparar la capacidad de los métodos de Kosko, BL y AHKBL. De estas pruebas resulto que existe una diferencia dramática entre Kosko y los otros dos esquemas. Sin embargo, no existe ninguna diferencia entre la capacidad presentada por la BAM con BL y la BAM con AHKBL. Lo anterior quiere decir que este último método es una codificación eficiente para una BAM en el sentido de la capacidad de almacenamiento.

También se realizaron pruebas para considerar la habilidad para la corrección de errores. Los resultados de estas pruebas se muestran en la figura B.17.

A parte de la capacidad de corrección de errores se registro el número de ciclos de aprendizaje requeridos por BL y AHKBL. Naturalmente, el número de ciclos de aprendizaje requeridos se incrementa cuando el número de patrones fundamentales es mayor. Aunque, en el AHKBL el número de ciclos de aprendizaje también se incrementa aún cuando el número de parámetros de aprendizaje es menor. Por ejemplo, con $n = p = 32$ y $N = 16$, el valor medio del número de ciclos de aprendizaje requeridos por el BL es de 12.75. En el caso del AHKBL con $\left(\rho_1 = 0.5, \rho_2 = \frac{1}{n} = 0.03125\right)$, el valor medio es de 51.11. Cuando se utilizan parámetros de aprendizaje mayores $\left(\rho_1 = 1, \rho_2 = \frac{2}{n} = 0.0625\right)$ el valor medio decrece a 10.23.

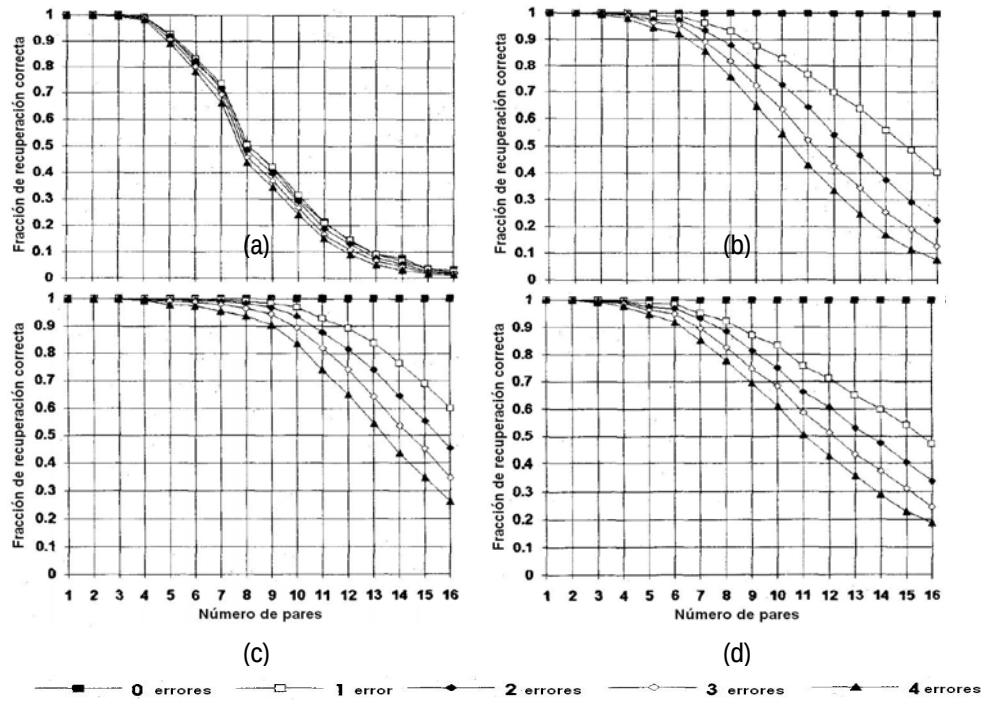



**Figura B.17** Comparación de la capacidad de corrección de errores de BAMs con dimensiones $n = p = 32$, entre tres métodos de codificación: (a) Kosko; (b) BL; (c) AHKBL con $\rho_1 = 0.5$ y $\rho_2 = 0.03125$ y (d) AHKBL con $\rho_1 = 1$ y $\rho_2 = 0.0625$.

## *Khorasani (1994)*

Las redes de memorias asociativas, generalmente, están caracterizadas por simples arquitecturas, sin embargo, existen varias desventajas que hacen su uso menos atractivo. Estas desventajas son: (i) baja capacidad de almacenamiento, (ii) dificultad en almacenar atractores altamente correlacionados, y (iii) un gran número de estados estables ficticios. Existen dos maneras básicas de superar esas desventajas. Un camino es cambiar la estructura de la red y otra es modificar el algoritmo de aprendizaje [14, 16, 31]. El método propuesto por Khorasani *et al* [43], del Departamento de Ingeniería Eléctrica y Computación en la Universidad Concordia, no está basado en la minimización de la función de la energía de los patrones fundamentales, sino que es una generalización de un algoritmo de aprendizaje autoasociativo desarrollado para redes de Hopfield.

El algoritmo de describe a continuación.

Para una red autoasociativa se dice que un vector $X_k$ = **(**$X_{k1}$, ..., $X_{kN}$**)** es estable si

$$\left( \sum_{j=1}^{N} W_{ij} X_{kj} - \theta_i \right) X_{ki} > 0 \text{, para } i = 1, ..., N$$

Entonces el problema de encontrar los pesos de la matriz **W** y los umbrales $\theta$, se reduce a resolver un sistema de desigualdades para cada patrón dado *k*. El algoritmo *Successive Over-Relaxation* (SOR) se utiliza para resolver estas desigualdades. SOR usa dos parámetros: (i) $\lambda$ - el factor de *over-relaxation* y (ii) $\xi$ - la constante de normalización. El factor $\lambda$ se elige entre 0 y 1, y el parámetro $\xi$ debe ser positivo. El efecto de $\xi$ es incrementar globalmente la magnitud de cada uno de los pesos y umbrales.

Ahora, se obtiene un algoritmo para una BAM heteroasociativa generalizando el algoritmo anterior utilizado para una red autoasociativa.

Similarmente, se propone que un par de vectores $X_k$ = **(**$X_{k1}$, ..., $X_{kN}$**)** y $Y_k$ = **(**$Y_{k1}$, ..., $Y_{kQ}$**)** es un estado estable si,

$$\left( \sum_{j=1}^{N} W_{ij} X_{kj} - \theta_i \right) Y_{ki} > 0 \text{, para } i = 1, ..., Q$$

Utilizando el sistema modificado de desigualdades el nuevo algoritmo es el siguiente.

Dado un patrón la red se inicializa de acuerdo a $\Delta W_j = 0$, para $j = 1, ..., N$ y $\Delta\theta = 0$. La salida de la *i*-ésima neurona se calcula como $S_i = \sum_{j=1}^{N} W_{ij} X_{pj} - \theta_i$, donde $W_{ij}$ es la memoria y tiene dimensiones de $Q$ x $N$, $\theta$ es el vector de umbrales y tiene dimensión 1 x $Q$. Después, la salida $S_i$ de la *i*-ésima neurona se multiplica por $Y_{pi}$. Si el producto es < $\xi$, entonces el ajuste de pesos se realiza de acuerdo a:

$$\Delta W_j = \Delta W_j - \frac{(1+\lambda)}{Q}\left[S_i - \xi Y_{pi}\right]X_{pj}, \qquad \text{para } j = 1, ..., N \tag{B.66}$$

de otra forma los pesos no cambian y se aplica el siguiente patrón como entrada a la red para el entrenamiento. El umbral también es ajustado para un patrón dado *p* como:

$$\Delta\theta = \Delta\theta + \frac{(1+\lambda)}{N}\left[S_i - \xi Y_{pi}\right] \tag{B.67}$$

Si todos los patrones han sido entrenados para la *i*-ésima neurona, entonces el ajuste del umbral es $\theta_i = \theta_i + \Delta\theta$, y el ajuste de los pesos debe ser $W_{ij} = W_{ij} + \Delta W$, para $j = 1, ..., N$. La matriz de los pesos utilizada pata la BAM modificada se obtiene de la transpuesta de *W* ($W^t$).

La fase de recuperación se realiza de forma síncrona. Se aplica como entrada el patrón *X* y la salida se calcula como $NetY = \text{sgn}\left(W^T X\right)$. Si la salida corresponde a un patrón fundamental *Y* se dice que la red se estabilizó, si no, se asigna *Y* = *NetY*, y se calcula $NetX = \text{sgn}(WY)$. Si esta salida corresponde a un patrón fundamental de entrada *X*, de otra forma, se asigna *X* = *NetX* y se procede a generar *NetY*. El proceso se repite hasta que la red se estabiliza.

Al comparar este algoritmo con el de Wang *et al* [31] se observa que es parecido debido a que los patrones se entrenan, pero en este algoritmo ya no se tiene el aumento *dummy*.

## *Wang C (1994)*

Wang y Tsai [44] de la Universidad de Kaohsiung en Taiwan, realizan el análisis de la capacidad de la BAM exponencial (EBAM) [30] cuando ésta es implementada con circuitos. Afirman que la BAM tiene la habilidad de corregir sus propios errores o que tiene tolerancia al fallo, es decir, se espera que dado un patrón que esté separado pocos bits del patrón deseado será recuperado correctamente. La razón por la cual se analiza la tolerancia al fallo se debe a que siempre existe algún ruido en los circuitos VLSI.

En este trabajo se utiliza la razón señal-a-ruido (SNR) para obtener el radio de atracción y la capacidad de la memoria.

Se parte de la ecuación de la SNR:

$$SNR_{eBAM} = \frac{2^{n-1}b^4}{2(M-1)(1+b^{-4})^{n-1}} \tag{B.68}$$

donde *n* es el mín(*n*, *p*). *n* y *p* son las dimensiones de los patrones de entrada y salida respectivamente.

Siendo $M$ el número de patrones entrenados y $r$ el radio de atracción dado, con $r = n/2$, la capacidad máxima para una EBAM para almacenar pares de patrones es:

$$M < 1 + \frac{2^{(n-2)}}{(n-r-1)\cdot b^{4(r-2)}} = M_{\max} \tag{B.69}$$

El radio de atracción puede ser estimado como la habilidad de tolerancia al fallo de la EBAM. Básicamente, si se deseara que esta habilidad mejorara, entonces inevitablemente la distancia de Hamming entre dos patrones almacenados debe incrementarse, por tanto la cantidad de patrones que se pueden almacenar debe decrecer. En consecuencia, la cantidad de patrones almacenados afecta la probabilidad de una recuperación correcta.

## *Xu (1994)*

Este trabajo presenta una memoria asociativa bidireccional asimétrica (*Asymmetric Bidirectional Asociativa Memory*, ABAM) propuesta por Xu *et al* [45] del Instituto de Matemáticas Computacionales y Aplicadas en Hong Kong. Esta red se construye con la estructura de la BAM de Kosko [14], pero con un esquema de codificación asimétrico para la matriz de conexión. Basado en la aplicación de un operador de interpolación específico, el nuevo esquema de codificación proporciona no sólo una generalización directa del trabajo de Kosko, sino que garantiza la recuperación de todos los patrones cuando éstos son linealmente independientes.

Dados los pares de patrones bipolares heteroasociativos

$$\{(X^{(1)}, Y^{(1)}), (X^{(2)}, Y^{(2)}), ..., (X^{(M)}, Y^{(M)})\}$$

donde $X^{(i)} \in \{-1,1\}^N$ y $Y^{(i)} \in \{-1,1\}^P$, para $i = 1, ..., M$, son los patrones en las dos poblaciones de neuronas $X$ y $Y$ respectivamente. Sean $Z^{(i)} = ((X^{(i)})^T, (Y^{(i)})^T)^T$ y $\{Z^{(1)}, Z^{(2)}, \ldots, Z^{(M)}\}$ los patrones autoasociativos en $\{-1,1\}^L$ donde $L = N + P$. Entonces la BAM es un sistema discreto en el tiempo que puede ser representado como un grafo direccional bipartido con $N + P$ nodos (neuronas) en $X \cup Y$, donde $X$ y $Y$ son conjuntos de nodos independientes (*i.e.*, no existen nodos interconectados del mismo conjunto). Sean los nodos fijos ($i$) y ($j$) en $X$ y $Y$ respectivamente, y sea

(a) $m_{ij}$ el peso de la arista ($i$, $j$);
(b) $\theta_i$ el umbral del nodo ($i$) y $\eta_j$ el umbral del nodo ($j$).

Las ecuaciones de evolución de la ABAM son las siguientes:

$$\mathbf{X}(t+1) = \text{sgn}(\mathbf{A}\mathbf{Y}(t) - \theta) \tag{B.70}$$

$$\mathbf{Y}(t+1) = \text{sgn}(\mathbf{B}\mathbf{X}(t) - \eta) \tag{B.71}$$

donde

$$\mathbf{A} = \mathbf{H}_M(\mathbf{X}, \mathbf{Y}), \qquad \mathbf{B} = \mathbf{H}_M(\mathbf{Y}, \mathbf{X}) \tag{B.72}$$

El modelo, en la fase de aprendizaje, pretende construir el operador de interpolación $\mathbf{H}_k(\mathbf{Y}, \mathbf{X})$ que debe cumplir la siguiente condición:

$$\mathbf{H}_k(\mathbf{Y}, \mathbf{X})X^{(i)} = Y^{(i)} \tag{B.73}$$

Este método de codificación asegura que cada patrón fundamental ($X^{(i)}$, $Y^{(i)}$) es un estado estable de la ABAM con los umbrales $\theta$ y $\eta$ iguales a cero.

La capacidad de almacenamiento de la ABAM no es menor que el mínimo de los rangos de los patrones fundamentales. El rango de un conjunto de vectores, **X**, es $r(\mathbf{X})$ y se define como el número de vectores linealmente independientes que existen en **X**.

Entonces, $\min\{r(\mathbf{X}), r(\mathbf{Y})\}$ es la capacidad de almacenamiento de la ABAM.

Se compararon la BAM, la IBAM, la memoria de Hopfield y la ABAM. La figura B.18 muestra la reconstruibilidad que es el número de estados estables que se pueden generar a partir de los patrones fundamentales.

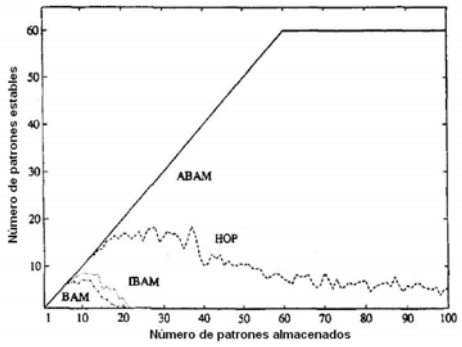



**Figura B.18** Gráfica de la comparación de la reconstruibilidad.

La comparación de la capacidad de corrección de errores entre las BAMs mencionadas anteriormente, se muestran en la figura B.19

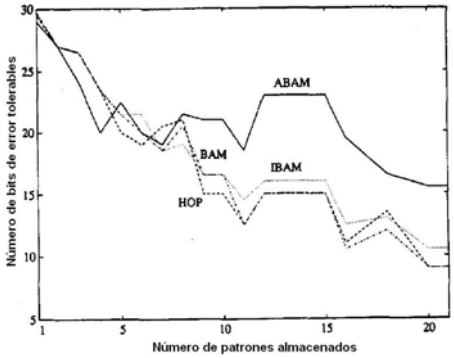



**Figura B.19** Gráfica de la capacidad de corrección de errores.

## *Hattori (1994)*

En este trabajo Hattori *et al* [46], de la Universidad de Keio en Japón, proponen un nuevo algoritmo para mejorar la capacidad de la BAM, el cual utiliza en la fase de aprendizaje dos etapas. En la primera etapa, se obtiene la matriz de correlación de la misma manera que Kosko [14]. En la segunda etapa, se utiliza el paradigma llamado *Pseudo-Relaxation Learning Algorithm* (PRLAB) [47] para BAM.

Considérese una *N*-*M* BAM con *N* neuronas en la primera capa y *M* neuronas en la segunda capa. Sea $W_{ij}$ la fuerza de conexión entre la *i*-ésima neurona de la primera capa y la *j*-ésima neurona de la segunda capa. Sea $\theta_{X_i}$ y $\theta_{Y_j}$ son los umbrales para la *i*-ésima neurona de la primera capa y la *j*-ésima neurona de la segunda capa, respectivamente. Sea $V = \{(X^{(k)},Y^{(k)})\}_{k = 1, ..., p}$ un conjunto de pares de patrones entrenados donde $X^{(k)} \in \{-1,1\}^N$ y $Y^{(k)} \in \{-1,1\}^M$.

Lo que realiza el PRLAB se describe a continuación.

Se dice que se garantiza que los vectores en *V* serán recuperados si el siguiente sistema lineal de desigualdades se satisface para toda $k = 1, ..., p$:

$$\left(\sum_{i=1}^{N} W_{ij} X_i^{(k)} - \theta_{Y_j}\right) Y_j^{(k)} > 0 \quad \forall j = 1, ..., M \tag{B.74}$$

$$\left(\sum_{j=1}^{M} W_{ij} Y_j^{(k)} - \theta_{X_i}\right) X_i^{(k)} > 0 \quad \forall i = 1, ..., N \tag{B.75}$$

PRLAB examina cada par entrenado ($X^{(k)}$, $Y^{(k)}$) uno por uno, sistemáticamente y cambia los pesos y los valores de los umbrales si las desigualdades anteriores no se satisfacen.

El método del PRLAB inicializa los valores de los pesos y de los umbrales de forma aleatoria.

La diferencia que tiene el PRLAB y el QLBAM es que este último, como primera etapa de la fase de aprendizaje, obtiene los pesos mediante la matriz de correlación (o matriz de Hebbian) como lo hace Kosko, y en la segunda etapa, utiliza el PRLAB tomando como valores iniciales los resultados de la primera etapa.

En la figura B.20 se muestra la gráfica para la comparación de la capacidad de almacenamiento, de los métodos utilizados por Kosko, PRLAB y QLBAM.

## *Leung (1994)*

Tomando la idea del *perceptron*, Leung [48], de la Universidad China de Hong Kong, propone el aprendizaje bidireccional (*Bidirectional Learning*, BL) para mejorar el

rendimiento de la recuperación de una BAM. Modificando la base de convergencia del *perceptron*, se puede probar que el BL provee una de las matrices de solución de conexión en un número finito de iteraciones (si la solución existe). De acuerdo con la convergencia del BL, la capacidad de la BAM con este aprendizaje es mayor o igual a aquellas con otras reglas de aprendizaje. Por lo tanto, el BL se considera como una regla de aprendizaje óptima para una BAM en el sentido de la capacidad.

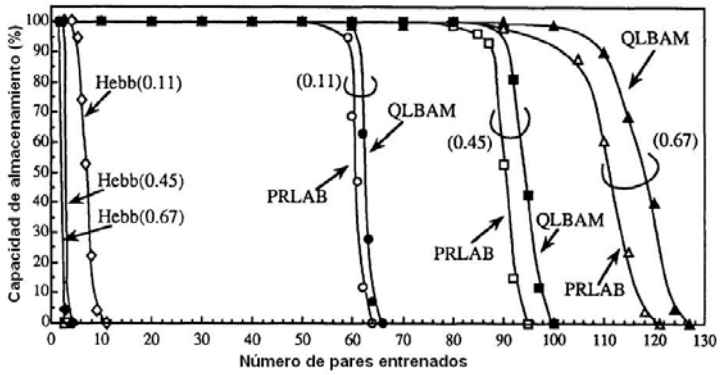



**Figura B.20** Gráfica de comparación de la capacidad de almacenamiento entre el método de Kosko, el PRLAB y el QLBAM. Los números en los paréntesis muestran la correlación de los diferentes pares.

El BL se describe a continuación.

La BAM almacena pares de vectores bipolares ($\mathbf{A}_k$, $\mathbf{B}_k$), $k$ = 1, ..., $N$, donde $\mathbf{A}_k = (a_{1k},...,a_{L_Ak})^T$, $\mathbf{B}_k = (b_{1k},...,b_{L_Ak})^T$, y $N$ es el número de pares de patrones de la biblioteca. Existen dos capas en la BAM. Una es la capa $f_A$ que tiene $L_A$ neuronas que constituyen el vector del patrón **A**. La otra capa $f_B$ tiene $L_B$ neuronas que constituyen el vector del patrón **B**.

Se desea que un punto fijo sea uno de los pares de patrones de la biblioteca. Un punto fijo ($\mathbf{A}_f$, $\mathbf{B}_f$) tiene las siguientes propiedades:

$$\mathbf{B}_f = \text{sgn}(\mathbf{M}\mathbf{A}_f) \quad \text{y} \quad \mathbf{A}_f = \text{sgn}\,(\mathbf{M}^T\mathbf{B}_f)$$

Se le llama matriz de solución de conexión a **M'** si cada patrón fundamental ($\mathbf{A}_k$, $\mathbf{B}_k$) es un punto fijo que satisface las siguientes dos ecuaciones,

$$\text{sgn}(\mathbf{M'}\mathbf{A}_k) = \mathbf{B}_k \tag{B.76}$$

y

$$\text{sgn}(\mathbf{M'}^T\mathbf{B}_k) = \mathbf{A}_k \tag{B.77}$$

donde $k$ = 1, …, $N$. Dado el patrón fundamental, el propósito de la regla de aprendizaje es encontrar una de las matrices solución de conexión (suponiendo que existe).

En BL, el conjunto fundamental es presentado repetida y secuencialmente a la BAM para actualizar la matriz de conexión de acuerdo a las ecuaciones anteriores hasta que todos los patrones fundamentales son correctamente clasificados por la matriz.

Leung prueba en este trabajo que si la matriz solución de conexión no existe, entonces cualquier otra regla de aprendizaje tampoco podrá encontrar una solución. Lo anterior demuestra que porqué el BL siempre tendrá igual o mejor desempeño que las otras BAM.

En la figura B.21 se muestran las graficas en donde se compara la habilidad de corrección de errores entre la BAM de Kosko [14] y el BL.

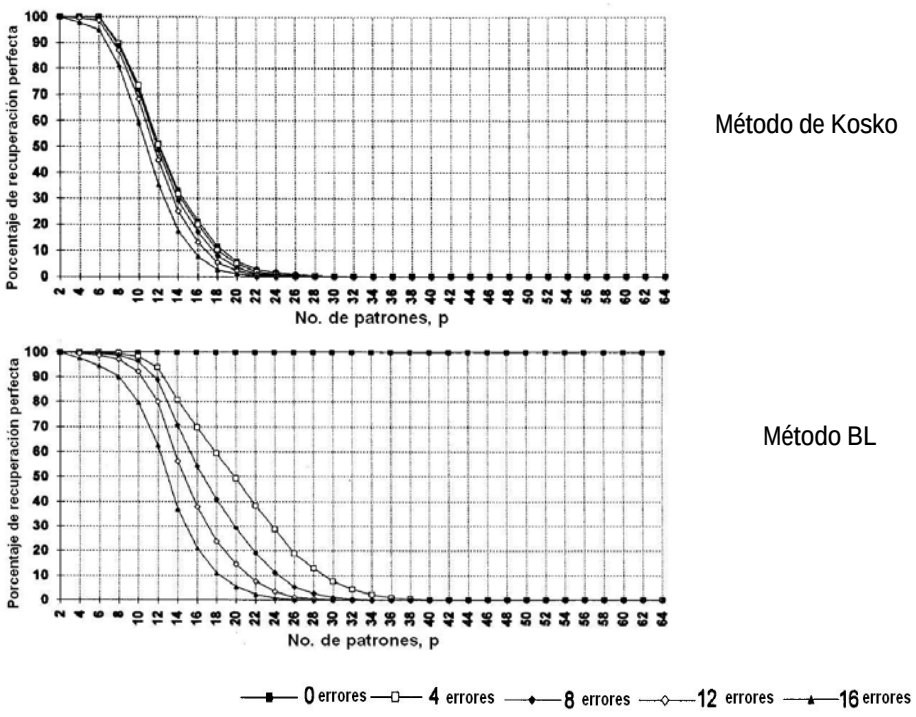



**Figura B.21** Graficas de la comparación de las habilidades de corrección de errores entre la BAM de Kosko y el BL.

## *Hu (1994)*

Basado en la aplicación de diferentes técnicas de ortogonalización para el entrenamiento de los patrones fundamentales, en este trabajo Hu *et al* [49] de la Universidad China de Hong Kong, proponen dos esquemas de codificación iterativos: POBAM (*Parallel Orthogonalization Based* BAM) y UOBAM (*Unilateral Orthogonalization Based* BAM).

Considere una BAM *N-P* con *N* neuronas en la primera capa y *P* neuronas en la segunda capa. Sean $A = (a_{ij})$ y $B = (b_{ij})$ las matrices de conexión entre la primera y la segunda capa.

Sea $\left\{\left(X^{(k)},Y^{(k)}\right)\right\}_{k=1}^{M}$ un conjunto dado de pares entrenados, donde $X^{(k)} \in \{-1,1\}^N$ y $Y^{(k)} \in \{-1,1\}^P$.

Primero se describe el método POBAM.

Se sabe que el método de codificación de Kosko [14] es:

$$A = \sum_{k=1}^{M}\left(X^{(k)}\right)\left(Y^{(k)}\right)^T, \quad B = A^T \tag{B.78}$$

El método de Kosko recupera el conjunto fundamental si los patrones fundamentales son ortogonales, si esto no sucede, entonces podrían ortogonalizarse mediante el procedimiento de ortogonalización de Gram.Schimit (GSO), sin embargo, aplicando este procedimiento a la codificación de Kosko resulta en un gran gasto computacional, por lo que, en el POBAM se sugiere una forma más económica:

$$A = \mathbf{H}_{\mathrm{M}}, \quad B = A^T \tag{B.79}$$

donde $\mathbf{H}_{\mathrm{M}}$ está dado por la siguiente fórmula recursiva:

$$\mathbf{H}_1 = \frac{\left(X^{(1)}\right)\left(Y^{(1)}\right)^T}{\left\|X^{(1)}\right\|\left\|Y^{(1)}\right\|}, \quad \mathbf{P}_1 = \frac{\left(X^{(1)}\right)\left(X^{(1)}\right)^T}{\left\|X^{(1)}\right\|^2}, \quad \mathbf{Q}_1 = \frac{\left(Y^{(1)}\right)\left(Y^{(1)}\right)^T}{\left\|Y^{(1)}\right\|^2}$$

$$\mathbf{H}_{k+1} = H_k + \frac{\varepsilon_k \eta_k^T}{\left\|\varepsilon_k\right\|\left\|\eta_k\right\|}, \quad \mathbf{P}_{k+1} = P_k + \frac{\varepsilon_k \varepsilon_k^T}{\left\|\varepsilon_k\right\|^2}, \quad \mathbf{Q}_{k+1} = Q_1 + \frac{\eta_k \eta_k^T}{\left\|\eta_k\right\|^2}, \quad k = 1, 2 \ldots, M\text{-}1 \tag{B.80}$$

donde $\varepsilon_k = X^{(k+1)} - P_k X^{(k+1)}$ y $\eta_k = Y^{(k+1)} - Q_k Y^{(k+1)}$

El POBAM aún tiene conexiones simétricas como Kosko.

Ahora se describe el UOBAM.

En este método, a diferencia del de Kosko**,** $B \neq \mathbf{M}^T$, sino que las ecuaciones son:

$$A = \mathbf{H}_{\mathrm{M}}(\mathbf{X}, \mathbf{Y}), \qquad B = \mathbf{H}_{\mathrm{M}}(\mathbf{Y}, \mathbf{X}) \tag{B.81}$$

Se puede observar de la ecuación anterior que la UOBAM tiene conexiones asimétricas.

En la figura B.22 se puede observar la gráfica de la comparación de las BAM descritas en este trabajo y las de Kosko, IBAM [33], ABAM [45] y Hopfield [13].

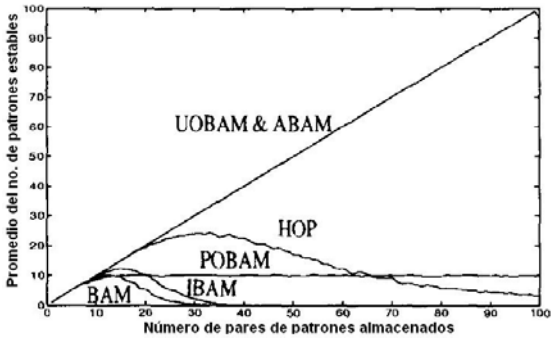



**Figura B.22** Comparación de la restaurabilidad entre la UOBAM, ABAM, POBAM, BAM de Kosko, IBAM y Hopfield.

La capacidad de corrección de errores de una red se determina mediante su habilidad de en la recuperación del estado estable original cuando ocurre un error en cualquiera de sus componentes.

La figura B.23 muestra la gráfica de comparación de la capacidad de corrección de errores entre los métodos antes mencionados.

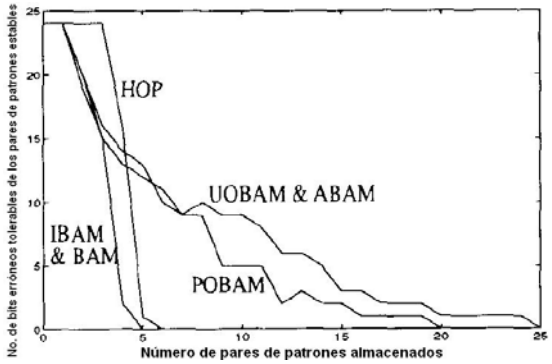



**Figura B.23** Gráfica de comparación de la capacidad de corrección de errores.

## *Wang Z (1996)*

Wang Z [50], del Instituto de Ingeniería de Sistemas de la Universidad Tianjin en la República de China, propone una matriz óptima en lugar de una matriz de correlación. Esta matriz óptima se determina mediante un aprendizaje de correlación simple que no requiere el cálculo de la pseudo inversa. Se presenta una versión lineal y otra no lineal, sin embargo, la introducción de características no lineales incrementa la habilidad de corrección de errores. Mediante este método se reducen ampliamente los estados espurios, lo que mejora ampliamente el desempeño de la BAM. Además, debido a que tiene conexiones asimétricas presenta mejores capacidad que otras BAMs.

En la tabla B.1 se muestra se muestran las capacidades de varios métodos.

**Tabla B.1 Capacidad de memoria para las BAM de Kosko, MT, PRLAB, LBAM y NBAM.**

| Número de neuronas | Capacidad (N, número de patrones entrenados) | | | | |
|---|---|---|---|---|---|
| | Kosko | Entrenamiento Múltiple | PRLAB | LBAM (Lineal) | NBAM (No Lineal) |
| 100-100 | 8 | 11 | 50 | 95 | 95 |
| 145-145 | 11 | 14 | 50 | 140 | 140 |
| 200-200 | 12 | 18 | 100 | 190 | 190 |
| 225-225 | 14 | 20 | 100 | 210 | 210 |

## *Sarkar (1996)*

En este trabajo Sarkar [51], de la Universidad de Miami, propone un método para calcular la cantidad exacta de ruido en la recuperación de patrones. Esto se puede utilizar para encontrar vectores que permitan construir matrices aumentadas como lo hacen los métodos de Wang *et al* [31] y Wang & Lee [35]. El cálculo del ruido también proporciona la dimensión mínima de los vectores aumentados. Las matrices construidas garantizan una recuperación perfecta, siempre que haya solución.

También se propone una BAM de tres capas (TLBAM) que requiere matrices de correlación de tamaño pequeño. El número de elementos de las matrices de correlación son proporcionales al número de patrones codificados.

En la figura B.24 se muestra la arquitectura de tres capas de la TLBAM.

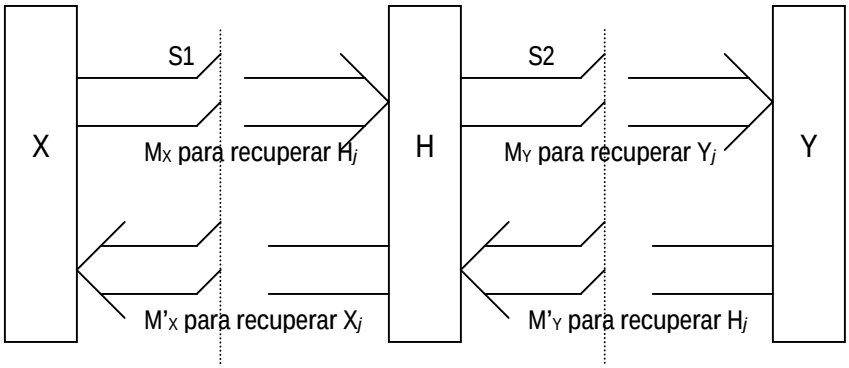



**Figura B.24** Diagrama a bloques de la TLBAM.

Se utiliza un nuevo conjunto de vectores bipolares, $H = \{H_1, H_2, ..., H_p\}$ para calcular las dos matrices de correlación, $M_A$ y $M_B$. La dimensión de los vectores en $H$ es múltiplo de 4; la dimensión se elige tal que es el número menor mayor o igual que $p$, el número de patrones. Se asume que los vectores $H_i$ son ortogonales entre sí.

Las matrices $M_A$ y $M_B$ se calculan como sigue:

$$M_A = \sum_{i=1}^{p} A_i^T H_i \quad \text{y} \quad M_B = \sum_{i=1}^{p} B_i^T H_i$$

Para recuperar un patrón $B_j$: i) Se cierra el interruptor S1, ii) Se abre el interruptor S2, y iii) El vector de entrada se aplica a la capa *A*. Se realiza el proceso de una BAM bidireccional hasta que se alcanza un estado estable, entonces la salida de la capa oculta *H* (digamos $H_f$) se registra. En el siguiente paso: i) Se abre el interruptor S1, ii) Se cierra el interruptor S2, y iii) $H_f$ se aplica a la capa oculta. La salida de la capa *B* es el patrón recuperado.

Se puede observar que si $H_f$ es uno de los vectores ortogonales que se utilizó para obtener las matrices de correlación $M_A$ y $M_B$, la recuperación del vector $B_j$ es un proceso de un paso y no requiere un proceso bidireccional. Sin embargo, para asegurar que $H_f$ es uno de los vectores codificados de *H*, la matriz de correlación $M_A$ puede necesitar ser aumentada. Para el aumento de esta matriz se puede utilizar el método utilizado por Wang *et al*.

Para la recuperación de $A_j$ se siguen pasos similares a los utilizados para recuperar $B_j$. Para una recuperación libre de ruido, puede ser necesario aumentar $M_B$.

## *Osana (1996)*

La memoria asociativa bidireccional caótica (*Chaotic Bidirectional Associative Memory*, CBAM) propuesta por Osana *et al* [52], de la Universidad de Keio en Japón, tiene las siguientes características:

- Puede manejar una a muchas asociaciones memorizando cada par entrenado junto con su información contextual.
- Tiene una estructura simple debido a que utiliza neuronas caóticas en la parte de la BAM convencional [14].
- Tiene una gran probabilidad de que todos los patrones deseados se recuperen.

La estructura de la CBAM se muestra en la figura B.25:

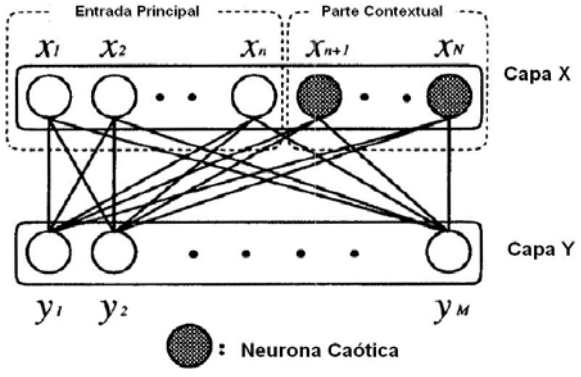



**Figura B.25** Estructura de la CBAM.

En la CBAM, cada patrón entrenado se memoriza junto con su propia información contextual. Ya que las neuronas caóticas utilizadas en parte de la red cambian sus estados mediante el caos, la CBAM puede recuperar patrones deseados plurales en forma dinámica.

Para la codificación, la CBAM es entrenada mediante una matriz de correlación como lo hace la BAM convencional:

$$W = \sum_{k=1}^{P} X^{(k)T} Y^{(k)} \tag{B.82}$$

donde $X^{(k)} \in \{-1,1\}^N$ y $Y^{(k)} \in \{-1,1\}^M$ y $P$ es el número de patrones entrenados que serán almacenados.

Al incluirle la información contextual al los patrones fundamentales de entrada, tenemos:

$$\{(X_1\_C_1, Y_1)\ (X_1\_C_2, Y_2)\ (X_2\_C_3, Y_3)\}$$

donde el subrayado muestra la información contextual. Aunque dicho método se utiliza para mejorar la capacidad de almacenamiento de la BAM, este método se aplica para memorizar relaciones de uno-a-muchos.

Para la decodificación se asume que la información contextual es desconocida por los usuarios, por lo que sólo es necesario proporcionarle un patrón de entrada.

Debido a que la CBAM está basada en la BAM, el proceso de recuperación es casi el mismo:

Paso 1. Se le entrega el patrón de entrada a la capa X.
Paso 2. El estado interno de la *j*-ésima neurona en la capa Y se calcula como:

$$u_j^y = \sum_{i=1}^{N} x_i(t) w_{ij} \tag{B.83}$$

La salida de la neurona es:

$$y_j(t) = f(u_j^y(t)) \tag{B.84}$$

donde

$$f(u) = \frac{1}{1+e^{(-u/\varepsilon)}} \tag{B.85}$$

donde ε es el parámetro de la pendiente.

Paso 3.1. Se calcula es estado interno de la *i*-ésima neurona de la parte principal y su salida es:

$$u_i^x(t+1) = \sum_{j=1}^{M} w_{ij} y_j(t) \quad (1 \le i < \text{n}) \tag{B.86}$$

$$x_i(t+1) = f(u_i^x(t+1)) \tag{B.87}$$

Paso 3.2 Debido a que las neuronas que corresponden a la información contextual son neuronas caóticas, el estado interno de la *i*-ésima neurona de la parte contextual es:

$$u_i^x(t+1) = \eta_i(t+1) + \zeta_i(t+1) \quad (n < i \le N) \tag{B.88}$$

donde

$$\eta_i(t+1) = k_m \eta_i(t) + \sum_{j=1}^{M} w_{ij} y_j(t) \tag{B.89}$$

$$\zeta_i(t+1) = k_r \zeta_i(t) - \alpha x_i(t) + a \tag{B.90}$$

La salida se calcula mediante la ecuación B.87.

Paso 4. Se repiten los pasos 2 y 3.

Debido a que este método es utilizado para hacer asociaciones de uno a muchos (diferente a lo que realizan otros métodos) no se muestra ninguna comparación.

## *Chen (1997)*

En este trabajo, Chen *et al* [53], de la Universidad de Nanjing de China, proponen una mejora de la BAM exponencial de Jeng [30]. Wang *et al* [35] propusieron una EBAM modificada (MEBAM) cuya ventaja sobre la EBAM era su gran capacidad de almacenamiento y la relajación en la suposición de continuidad [14], sin embargo en su trabajo no se ilustra la mejora de la capacidad de corrección de errores.

Se asume que existen *M* pares de datos almacenados ($X_i$, $Y_i$), ($i$ = 1, 2, ..., *M*), donde $X_i \in \{-1,1\}^n$ y $Y_i \in \{-1,1\}^p$.
La suposición de continuidad es:

$$\frac{H(X_i, X_j)}{n} \approx \frac{H(Y_i, Y_j)}{p} \tag{B.91}$$

donde H(...) denota la distancia de Hamming. Básicamente, cuando dos entradas casi idénticas ($X_i$ y $X_j$, $i \neq j$) son asociadas a dos salidas casi distintas ($Y_i$ y $Y_j$, $i \neq j$) o viceversa, la EBAM puede confundirse y converger a otro patrón.

Las ecuaciones de la regla de actualización de la IEBAM (*Improved Exponential* BAM) son:

$$Y' = \operatorname{sgn}\left( \sum_{i=1}^{m} Y_i b^{(\langle X_i, X \rangle + \langle Y_i, Y \rangle)} \right) \tag{B.92}$$

$$X' = \operatorname{sgn}\left( \sum_{i=1}^{m} X_i b^{(\langle X_i, X \rangle + \langle Y_i, Y \rangle)} \right) \tag{B.93}$$

Las ecuaciones anteriores permiten una mejor capacidad de almacenamiento y una mejor habilidad para la corrección de errores que la MEBAM.

La figura B.26 muestra las capacidades de almacenamiento que se miden por el número de recuperaciones exitosas de entre *M* pares entrenados generados aleatoriamente. Al mismo tiempo, las capacidades de error de corrección se prueban invirtiendo *r* bits aleatoriamente. En este trabajo se utiliza b = *e*.

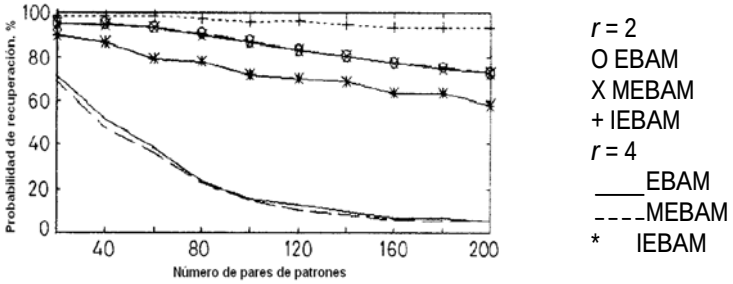



**Figura B.26** Gráfica de la comparación de las capacidades de almacenamiento y corrección de errores.

## *Haryono (1997)*

En el Departamento de la Ciencia de la Computación en Información en Tailandia, Haryono *et al* [54], proponen un método llamado BOAM (*Bipolar-Orthogonal Augmentation Method*) para mejorar la capacidad de almacenamiento.

Las siguientes propiedades sirven de base para este método:

a) Entre más grande es la dimensión de los patrones fundamentales (*n* y *p*), mayor es el número de patrones que se pueden recuperar.
b) Cuando los patrones de entrada son ortogonales al igual que los patrones de salida, todos los patrones del conjunto fundamental pueden ser recuperados.
c) Cuando $X_i = Y_i$, y por tanto, $n = p$, la BAM puede almacenar y recuperar los $2^n$ estados disponibles de los pares de patrones, si se codifican en forma bipolar.
d) La densidad de los patrones de entrada, esto es, la razón entre el número de bits "1" y la dimensión de los patrones, es un factor de influencia para la capacidad de la BAM.

Wang *et al* [31] proponen una aumento-ortogonal de elementos *dummy* que son binarios, aún cuando los patrones entrenados son bipolares. En este trabajo también se propone un

aumento pero con elementos bipolares. Este cambio de elementos permite que la capacidad de almacenamiento sea mayor.

## *Araújo (1997)*

En este trabajo, Araújo y Haga [55], de la Universidad de Sao Paulo del Departamento de Ingeniería Eléctrica, presentan dos estrategias para mejorar el rendimiento de la BAM. El USA (*Unlearning of Spurious Attractors*) consiste en disociar cualquier estímulo de una respuesta incorrecta. La BDR (*Bidirectional Delta Rule*) se extiende para el uso en BAMs. Estos paradigmas no demandan un preprocesamiento de las entradas, entrenan la red rápidamente, tienen un comportamiento estable y presentan alta tolerancia al ruido.

La BAM [14] es un modelo de memoria que asocia pares de patrones. La BAM está compuesta de dos capas las cuales están totalmente conectadas en forma bidireccional. La capa de entrada tiene *m* unidades de procesamiento, $X^{(k)} = \left\{x_1^k, x_2^k, ..., x_m^k\right\}$ para $k = 1, ..., p$, mientras que la capa de salida tiene *n* unidades de procesamiento $Y^{(k)} = \left\{y_1^k, y_2^k, ..., y_n^k\right\}$ para $k = 1, ..., p$.

Una matriz de ponderación **W** de *m* x *n*, que es la matriz de correlación, y que se construye de la siguiente forma:

$$W = \sum_{k=1}^{p} X_k^T Y_k$$

donde *p* es el número total de patrones.

Primero se utiliza el algoritmo *Pseudo-Relaxation Learning* (PRLAB), para encontrar los pesos de las conexiones y los umbrales de las unidades. Entonces, se construye la matriz de correlación como una BAM. Posteriormente, para ir desvaneciendo los errores, los pesos y los umbrales se varían de la siguiente forma: para las unidades de entrada,

$$\Delta W_{ij} = \frac{\lambda}{1+m}\left[S_{x_i}^{(k)} - \xi X_i^{(k)}\right]Y_i^{(k)} \quad \text{si} \quad S_{x_i}^{(k)} X_i^{(k)} \le 0$$

$$\Delta T_{X_i} = \frac{\lambda}{1+m}\left[S_{x_i}^{(k)} - \xi X_i^{(k)}\right] \qquad \text{si} \quad S_{x_i}^{(k)} X_i^{(k)} \le 0$$

Para la unidades de salida:

$$\Delta W_{ij} = \frac{\lambda}{1+n}\left[S_{Y_j}^{(k)} - \xi Y_j^{(k)}\right]X_i^{(k)} \quad \text{si} \quad S_{Y_j}^{(k)} Y_j^{(k)} \le 0$$

$$\Delta T_{Y_j} = \frac{\lambda}{1+n}\left[S_{Y_j}^{(k)} - \xi Y_j^{(k)}\right] \qquad \text{si} \quad S_{Y_j}^{(k)} Y_j^{(k)} \le 0$$

donde $S_{X_i}^{(k)} = \sum_{j=1}^{m} W_{ij} Y_j^{(k)} - T_{X_i}$ , $S_{Y_j}^{(k)} = \sum_{i=1}^{n} W_{ij} X_i^{(k)} - T_{Y_j}$ y λ es un factor de relajación; ξ=0.1.

En el USA se tiene una etapa de entrenamiento supervisado en el cual el producto externo de cada resultado espurio se sustrae de la matriz de correlación. Por lo tanto, mientras los atractores espurios se presenten, **W** se varía para cada patrón *k* como sigue:

$$\Delta W_k = -\left(\frac{\lambda}{m+1}\right) \times \left((S^{(k)}\right)^T R^{(k)} \tag{B.94}$$

donde $S^{(k)}$ y $R^{(k)}$ son atractores espurios y $\frac{\lambda}{m+1}$ es la razón de *unlearning*.

Los mínimos locales que producen los atractores espurios se desvanecen a cada paso del *unlearning*. Por lo tanto, los atractores deseados pueden reaparecer.

La BDR presupone que los pesos son actualizados bidireccionalmente. Por lo tanto, la BDR actualiza la matriz de correlación, mientras existan atractores espurios, cuando la información fluye en forma directa (entrada-salida) o hacia atrás (salida-entrada). La actualización está basada en la regla de Widrow-Hoff [56], como sigue.

Para el flujo de información entrada-salida:

$$\Delta W_{ij} = \frac{\lambda}{m+1}\left(X_i^{(k)} - S_i^{(k)}\right) Y_j^{(k)} \tag{B.95}$$

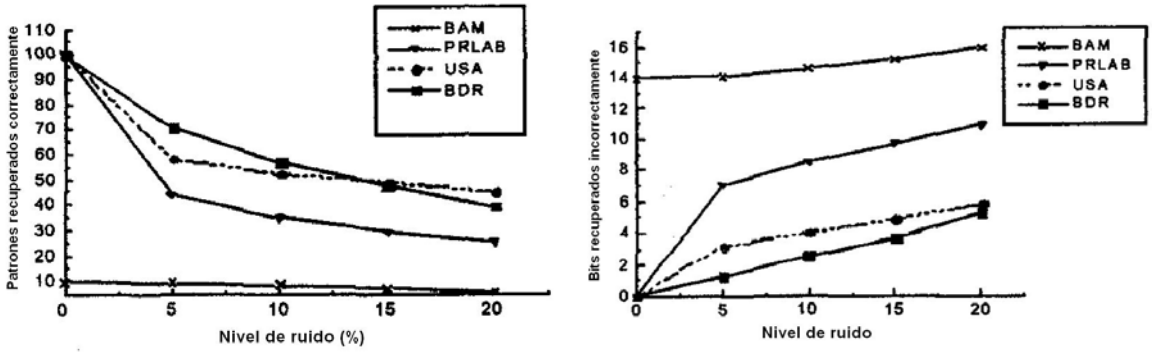
Para el flujo de información salida-entrada:

$$\Delta W_{ij} = \frac{\lambda}{n+1}\left(Y_j^{(k)} - R_j^{(k)}\right) X_i^{(k)} \tag{B.96}$$

donde $\frac{\lambda}{q+1}$ es la razón *unleraning* para $q = n$ o $q = m$; y $\left(X_i^{(k)}, Y_j^{(k)}\right)$ es el objetivo.

La memoria es entrenada con los 10 siguientes patrones de dimensión 9 x 9.

La figura B.27 muestra la comparación el porcentaje del número de patrones recuperados cuando se le agrega un cierto ruido a los patrones entrenados, de tres métodos el PRLAB (*Pseudo-Relaxation Learning Algorithm*), la USA y la BDR.




(a) (b)

**Figura B.27** Gráfica de comparación entre el PRLAB, la USA y la BDR. (a) Recuperaciones correctas y (b) Bits erróneos contra nivel de ruido en la entrada.

## *Ritter (1999)*

Supóngase que se desea retroalimentar síncronamente la salida $y_1^\xi = f\left(M \times x^\xi\right)$ a una memoria asociativa para mejorar la exactitud de la recuperación. En la BAM clásica [14] el esquema más simple de retroalimentación es pasar $y_1^\xi$ a través de **M**' (transpuesta de **M**). Si el patrón resultante $x_1^\xi = f\left(M' \times y_1^\xi\right)$ se retroalimenta a través de M, se obtiene un nuevo patrón $y_2^\xi$, el cual puede ser alimentado hacia M' para producir $x_2^\xi$, y así sucesivamente. Este ir y venir resonará en un par fijo $\left(x_f^\xi, y_f^\xi\right)$. Sin embargo, las BAM clásicas, siendo generalizaciones de la memoria de Hopfield, tienen limitaciones similares a ésta. El número de asociaciones que pueden ser almacenadas en la memoria y recuperadas en forma efectiva es muy limitada.

A continuación se describen las memorias asociativas morfológicas bidireccionales (*Morphological Bidirectional Associative Memory*, MBAM) [57], que son sorprendentemente muy similares a las clásicas, sin embargo tienen distinto comportamiento.

La teoría de las memorias morfológicas se describe en la sección 2.2.5.

Recordando: $$M = \bigvee_{\xi=1}^{k} \left(y^\xi \wedge (-x^\xi)'\right)$$

En el esquema de retroalimentación, la MBAM emplea la memoria conjugada o dual **M*** de **M**. La *ij*-ésima entrada de la matriz **M*** está dada por:

$$m_{ij}^* = -m_{ij} = -\bigvee_{\xi=1}^{k} (y_j^\xi - x_i^\xi) = \bigwedge_{\xi=1}^{k} \left(x_i^\xi - y_j^\xi\right)$$

Mientras la memoria M recupera el patrón $\mathbf{y}^\xi$ cuando se le presenta el patrón $\mathbf{x}^\xi$, la memoria dual M* recupera el patrón $\mathbf{x}^\xi$ cuando se le presenta el patrón $\mathbf{y}^\xi$.

## *Wu (2000)*

Wu y Pados [58], de la Universidad Estatal de Nueva York, proponen un método diferente para desarrollar una memoria asociativa bidireccional *feedforward*. El procedimiento es conceptualmente simple y de implementación directa.

Dado un conjunto de $p$ pares de prototipos bipolares bidireccionales $\{(\mathbf{x}^{(1)},\mathbf{y}^{(1)}), (\mathbf{x}^{(2)},\mathbf{y}^{(2)}), ..., (\mathbf{x}^{(p)},\mathbf{y}^{(p)})\}$, $\mathbf{x}^{(i)} \in \{-1,1\}^n$, $\mathbf{y}^{(i)} \in \{-1,1\}^m$, $\mathbf{x}^{(i)} \neq \mathbf{x}^{(j)}$ y $\mathbf{y}^{(i)} \neq \mathbf{y}^{(j)}$ para $i \neq j$, se diseñan dos distintas BAM de dos capas *feedforward*, con un límite rígido de neuronas de McCulloch-Pitts, que implementan las asociaciones bidireccionales $\mathbf{x}^{(i)} \leftrightarrow e_i$ y $y^i \leftrightarrow e_i$, donde $\{e_1, e_2, ..., e_p\}$ son los vectores unidad estándar base $R^p$. Entonces se combinan las dos previas BAM para implementar las asociación deseada $\mathbf{x}^{(i)} \leftrightarrow y^{(i)}$ en la forma de una sola red feedforward de tres capas (Ver figura B.28). El proceso de diseño $\mathbf{x}^{(i)} \leftrightarrow e_i$ (o $y^{(i)} \leftrightarrow e_i$) es un algoritmo *one-shot*. El máximo número de patrones fundamentales que pueden almacenarse (capacidad de almacenamiento) es $2^{\min(n,m)}$. Los patrones fundamentales pueden estar arbitrariamente correlacionados a diferencia de la ABAM [45].

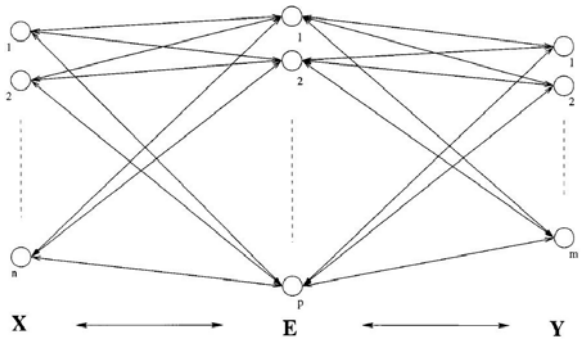



**Figura B.28** BAM de tres capas.

Durante la etapa de aprendizaje, la primera capa es responsable de construir un clasificador de la distancia mínima de Hamming, que está formado por agrupaciones que no se sobreponen unas con otras y que cubren el espacio de entrada. Esto se logra implementado un mapeo (asociación) del espacio de entrada hacia un conjunto de vectores intermediarios, E. La segunda capa implementa un mapeo del conjunto de vectores intermediarios hacia los patrones de salida. En la etapa operacional, la primera capa clasifica un vector de entrada hacia una de las agrupaciones de acuerdo con el criterio de la distancia mínima de Hamming. La recuperación final (asociación o correspondencia) se completa mediante la segunda capa cuando el patrón de salida es recuperado.

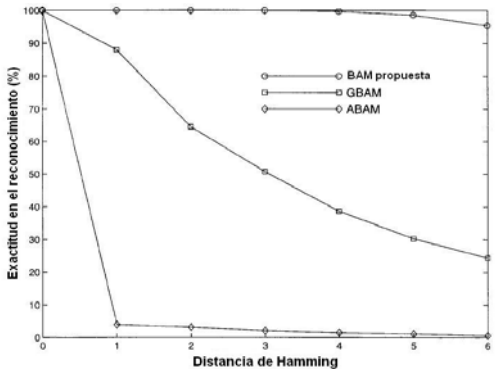



**Figura B.29** Comparación de la exactitud en el reconocimiento entre la GBAM, la ABAM y la BAM propuesta en este trabajo.

## *Lenze (2001)*

Leung [48] introdujo la regla de aprendizaje basada en la idea del *perceptron* para mejorar el rendimiento de la recuperación en memorias asociativas bidireccionales. Sin embargo, la solución existía siempre y cuando el conjunto fundamental fuera fuertemente lineal separable por hiperplanos a través del origen.

Este trabajo Lenze [59], de la Universidad de Ciencias Aplicadas en Alemania, se basa en la idea de Leung pero utilizando dilatación y traslación de manera que los patrones fundamentales estén separados por hiperplanos que no contengan el origen.

El método se describe a continuación.

Sean $n$, $m \in$ N dados arbitrariamente. La BAM generalizada puede tener $n$ neuronas de entrada, $m$ neuronas de salida, y $nm$ pesos de conexión $w_{ij} \in$ R, $1 \le i \le n$, $1 \le j \le m$. Más aún, puede tener $n$ pesos de dilatación y traslación de entrada $\alpha_i, d_i \in$ R, $\alpha_i > 0$, $1 \le i \le n$, y m pesos de dilatación y traslación de salida $\beta_j, f_j \in$ R, $\beta_j > 0$, $1 \le j \le m$.

El problema estándar en el diseño de una BAM es almacenar un conjunto de $t$ patrones codificados bipolares,

$$\left(\vec{x}^{(s)}, \vec{y}^{(s)}\right) \in \{-1,1\}^n \times \{-1,1\}^m, \quad 1 \le s \le t$$

Entonces, sean $n$, $m \in \mathbf{N}$ dados arbitrariamente y sea la BAM generalizada que puede tener $n$ neuronas de entrada y $m$ neuronas de salida, se asume que el conjunto anterior se extiende formalmente a un conjunto de muchas asociaciones:

$$\left(\vec{x}^{(s)}, \vec{y}^{(s)}\right) = \left(\vec{x}^{(s-1)(\mathrm{mod}\, t)+1}, \vec{y}^{(s-1)(\mathrm{mod}\, t)+1}\right) \tag{B.97}$$

Para $\vec{\alpha}, \vec{d} \in \mathbf{R}^n$ con $\vec{\alpha} > \vec{0}$ y $|\vec{d}| < \vec{\alpha}$, también $\boldsymbol{\beta}, \mathbf{f} \in \mathbf{R}^m$ con $\boldsymbol{\beta} > 0$ y $|\mathbf{f}| < \boldsymbol{\beta}$, y , finalmente, $\mathbf{w}^0 \in \mathbf{R}^{nm}$ dados arbitrariamente, el esquema generalizado de aprendizaje está definido recursivamente por:

$$\bar{y}_j^s = \operatorname{sgn}\left(\sum_{i=1}^{n} w_{ij}^{s-1}\left(\alpha_i x_i^s - d_i\right)\right), \qquad \bar{x}_i^s = \operatorname{sgn}\left(\sum_{j=1}^{m} w_{ij}^{s-1}\left(\beta_j y_j^s - f_j\right)\right)$$

$$\Delta_1 w_{ij}^s = \left(y_j^s - \bar{y}_j^s\right)\left(\alpha_i x_i^s - d_i\right), \qquad \Delta_2 w_{ij}^s = \left(x_i^s - \bar{x}_i^s\right)\left(\beta_j y_j^s - f_j\right)$$

$$w_{ij}^s = w_{ij}^{s-1} + \Delta_1 w_{ij}^s + \Delta_2 w_{ij}^s$$

## *Eom (2001)*

La memoria asociativa bidireccional propuesta por Jeng *et al* [30] ha sido utilizada ampliamente para aplicaciones como reconocimiento de patrones y compresión de datos debido a su estabilidad, capacidad de almacenamiento exponencial y a su gran habilidad para la corrección de errores.

Eom *et al* [60], del Instituto Avanzado de Ciencia y Tecnología en Corea, presentan una ecuación para determinar el radix (la base) que permita una recuperación perfecta utilizando la información de la distancia mínima obtenida estadística y deterministamente.

Se asume que existen *p* pares de patrones almacenados , donde $\mathbf{x}^i \in \{-1, 1\}^n$ y $\mathbf{y}^i \in \{-1, 1\}^m$. Cada elemento bipolar de los patrones se genera aleatoriamente con igual probabilidad. El proceso bidireccional de recuperación de la EBAM en un ciclo es:

$$y = f(x) = \operatorname{sgn}\left(\sum_{i=1}^{p} y^i b^{(x^i)^T x}\right) \tag{B.98}$$

$$x' = g(y) = \operatorname{sgn}\left(\sum_{i=1}^{p} x^i b^{(y^i)^T y}\right) \tag{B.99}$$

A partir de estas dos ecuaciones se determina el radix que garantiza una atracción directa dentro del radio *r* obtenida de la distancia mínima de Hamming:

$$b = (p-1)^{\frac{1}{2(s_{\min}-2r)}} \tag{B.100}$$

donde $s_{min} = \min_{\forall h} s_h$, $s_h = \min_{\forall i \neq h} d_H(\mathbf{x}^i, \mathbf{x}^h)$ y $r = s_{min}/2$. Más aún, eligiendo $b = (p\text{-}1)^{1/2}$, se puede garantizar que todos los patrones ruidosos con $r < s_{min}/2$ son directamente atraídos sin la necesidad de la información de $s_{min}$.

Sin embargo, si no se cuenta con la información de la distancia, entonces, se puede utilizar el valor esperado. Usando la probabilidad:

$$\Pr(s_h = r) = \Pr(s_h \geq r) - \Pr(s_h \geq r + 1) \tag{B.101}$$

el valor esperado de la distancia mínima de Hamming es, aproximadamente:

$$n - (p - 1)(1 - 2^{-n}) \tag{B.102}$$

Sustituyendo uno de estos valores esperados en $s_{min}$ de la ecuación B.100 es, estadísticamente, una solución factible, aunque la recuperación de los patrones fundamentales no está determinada estadísticamente.

Si el método falla al atraer ciertos patrones fundamentales, los errores se pueden compensar mediante el siguiente algoritmo de aprendizaje.

Si se define $u^h = \left[b^{(x^1)^t x^h}, ..., b^{(x^p)^t x^h}\right]$, entonces:

$$y_k^h \lambda_k^T u^h > 0 \quad \forall h, \forall k \tag{B.103}$$

donde $\lambda_k = \left[y_k^1, ..., y_k^p\right]$ son los pesos aprendidos. Las desigualdades de la ecuación anterior pueden resolverse con el método de pseudo-relajación (*Pseudo-Relaxation Algorithm*, PRA) propuesto por Oh y Kothari [61] para obtener los pesos.

En este algoritmo, el radix puede ser determinado por la capacidad estadística de la EBAM, $p \approx 2^n/(1 + b^{-4})^n$.

La figura B.30 muestra el efecto que tienen los pesos aprendidos sobre la EBAM, además de su capacidad estadística. Las simulaciones se realizaron para $n = m = 8$. Los resultados se promediaron sobre 100 conjuntos de pares de patrones seleccionados aleatoriamente. Durante el proceso de aprendizaje, el radix se incremento 0.01 comenzando desde 1.01.

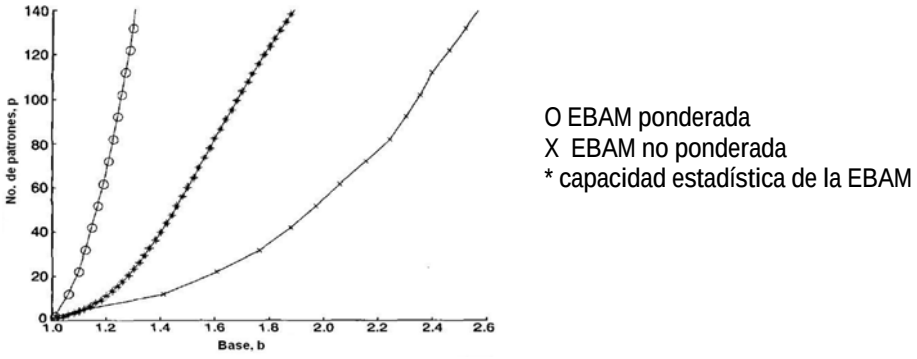



**Figura B.30** Número de patrones recuperados contra el radix.

## *Lee (2004)*

La simetría de las interconexiones entorpece el rendimiento de las BAMs en el almacenamiento de patrones y en su habilidad de recuperación. Para desahogar estas limitaciones se han propuesto métodos como la BAM asimétrica (ABAM) [45] y la BAM general (GBAM) propuesta por Shi *et al* [62]. Sin embargo la ABAM requiere que sus patrones fundamentales tengan una independencia lineal mientras que la GBAM sólo necesita separación lineal de sus patrones.

Shi *et al* definen la bola unidad de Hamming alrededor de cada patrón almacenado y proponen una regla de aprendizaje tal que todos los vectores que pertenecen a las bolas de Hamming son bidireccionalmente separables en forma lineal mediante las matrices de conexión de las dos capas (**A** y **B**).

En este trabajo, Lee y Chuang [63], de la Universidad de Vanung en Taiwán, proponen ampliar los tamaños de las bolas de Hamming hasta que sean linealmente no separables. Para esto se utiliza un algoritmo de mínima superposición que proporciona el aprendizaje de los pesos de conexión.

En la figura B.31 se muestran las gráficas de comparación entre la GBAM y la GBAM modificada propuesta en este trabajo (MGBAM).

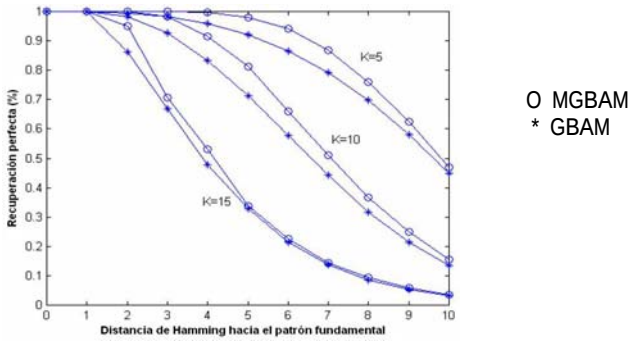



**Figura B.31** Probabilidad de recuperación contra distancia de Hamming.

## *Zheng (2005)*

Zheng *et al* [64], de la Universidad Wuhan en China, proponen la combinación de dos métodos que permitan mejorar la capacidad de la BAM.

El primer método es el de aumentación *dummy*, el cual permite la corrección de los patrones recuperados [31].

Por otra parte, también se utiliza el algoritmo del gradiente descendiente óptimo que posibilita en agrandamiento del radio de atracción de los patrones fundamentales para que se tenga una mejor convergencia hacia cada uno de los patrones fundamentales.

En la figura B.32 se muestra la gráfica de comparación de la capacidad de almaacenamiento entre la BAM de Kosko [14], OGDA (*Optimal Gradient Descent Algorithm*) [39] y el propuesto en este trabajo.

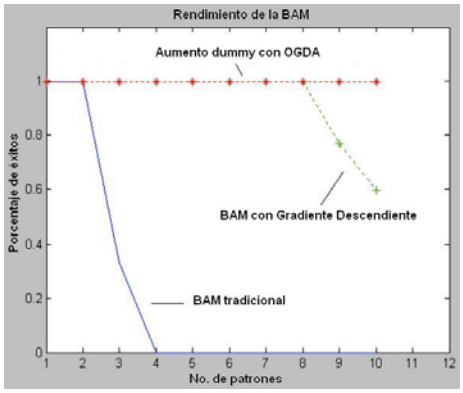



**Figura B.32** Pruebas de la capacidad de almacenamiento de tres diseños de BAM.

## *Shen (2005)*

En este trabajo, Shen y Cruz [18], de la Universidad Estatal de Ohio, utilizan una ponderación optimizada para la matriz de correlación. Dado un conjunto de pares entrenados, se determinan los pesos para estos pares en la matriz de correlación lo que resulta en un conjunto de máxima tolerancia al ruido.

Dado un conjunto de pares entrenados $\{(X_i, Y_i)\}_{i=1}^{N}$, la matriz de correlación ponderada es

$$M = \sum_{i=1}^{N} w_i X_i^T Y_i \qquad \text{(B.104)}$$

donde

$$X_i = (x_{i1}, x_{i2}, ..., x_{iQ})$$
$$Y_i = (y_{i1}, y_{i2}, ..., y_{iP})$$

$Q$ y $P$ son las longitudes de los patrones de entrada y salida, respectivamente. $W = (w_1, w_2, ..., w_N)$ es el vector de pesos entrenados.

Se utiliza un algoritmo genético estándar para calcular los pesos que maximicen la función objetivo.

Cualquier par de entrada ruidoso que esté dentro del conjunto de tolerancia convergirá al patrón fundamental correcto. Shen y Cruz proponen un límite superior para saber cuáles patrones, con cierta distancia de Hamming, se podrán recuperar:

$$\hat{F} = \frac{1}{2}\min\left( \min_{0\le i\ne j\le N} H_x(X_i, X), \min_{0\le i\ne j\le N} H_y(Y_i, Y) \right) + 1$$

En la gráfica B.33 se muestra el costo computacional requerido para los valores que puede tomar $\hat{F}$ .

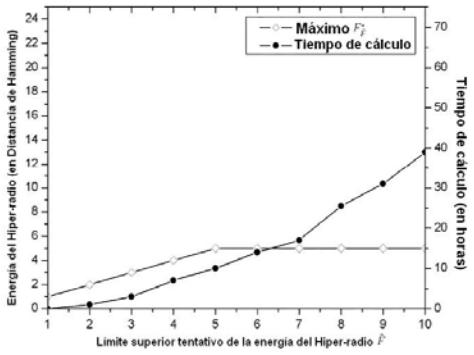



**Figura B.33** Máximo $\hat{F}$ versus tiempo de cálculo

De la gráfica anterior se puede observar que el costo computacional del algoritmo utilizado en este trabajo es muy grande. Conforme $\hat{F}$ aumenta, el tiempo de cálculo también aumenta. Cabe hacer notar que el tiempo de cálculo está dado en horas.

Para las simulaciones se utilizaron los siguientes patrones. La dimensión de los patrones de entrada es de 288, mientras que la dimensión de los patrones de salida es de 280.


Para esta simulación se calcula el límite superior:

$$H_x(X_1, X_2) = 90, \quad H_y(Y_1, Y_2) = 116$$
$$H_x(X_1, X_3) = 100, \quad H_y(Y_1, Y_3) = 134$$
$$H_x(X_2, X_3) = 50, \quad H_y(Y_2, Y_3) = 128$$
$$\hat{F} = \frac{50}{2} + 1 = 26$$

En este caso, el límite máximo es 26. Sin embargo se puede utilizar un número menor a éste. Para la simulación se encuentra $F^* = 5$. Regresando a la figura B.34 podemos observar que para este valor de 5, el tiempo de cálculo es aproximadamente de 15 horas.

En esta simulación se generaron 10000 muestras tomando como base cada uno de los patrones de entrada (las figuras). Todos los patrones se recuperan de forma correcta; sin

embargo, de los para patrones de entrada ruidosos, sólo se recuperaron aquellos patrones cuya distancia de Hamming era menor de 4 con respecto a los patrones originales. Los patrones con una distancia de Hamming mayor a 5 no se recuperan correctamente.

Cabe hacer notar que, finalmente, son sólo tres parejas de patrones las que se entrenaron con esta BAM.

## *Chartier (2006)*

En este trabajo, Chartier y Boukadoum [65], de la Universidad de Québec en Montreal, presentan un nuevo modelo de memoria asociativa bidireccional que permite el aprendizaje tanto de patrones binarios como de patrones en escala de grises.

Las propiedades que presenta este modelo son:

- Aprendizaje en línea
- Desarrolla iterativamente los pesos de las conexiones que convergen a una solución óptima, local y estable, incorporando la retroalimentación no lineal utilizada en la función de salida.
- El aprendizaje se basa solamente en la correlación de los patrones sin la necesidad de una capa escondida.
- La regla de aprendizaje es simple y permite asociar patrones tanto de dimensiones distintas como patrones de las mismas dimensiones.
- La función de salida, junto con la regla de aprendizaje, se elige de manera que la matriz de pesos no sólo desarrolla puntos atractores fijos para patrones bipolares sino también para patrones con valores reales.

La función de salida utilizada en este modelo está basada en la ecuación clásica de Verhulst:

$$\frac{dz}{dt} = G(1-z)z \tag{B.105}$$

donde $G$ es un parámetro general. La ecuación B.105 tiene dos puntos fijos, $z = 0$ y $z = 1$. Sin embargo, sólo $z = 1$ es un punto estable fijo y por lo tanto (2.3.112) tiene un solo atractor. Esta ecuación se modifica de manera que existan dos atractores:

$$\frac{dz}{dt} = G(1-z^2)z \tag{B.106}$$

La ecuación resultante tiene tres puntos fijos, -1, 0 y 1, de los cuales sólo $z = -1$ y $z = 1$ son puntos estables fijos.

La función de salida final para ambas direcciones está expresada por las siguientes ecuaciones:

$$\forall i, \ldots, N,\ \mathbf{y}_{i[t+1]} = f(\mathbf{a}_{i[t]}) = \begin{cases} 1, & \text{si } \mathbf{a}_{i[t]} > 1 \\ -1, & \text{si } \mathbf{a}_{i[t]} > -1 \\ (\delta+1)\mathbf{a}_{i[t]} - \delta\mathbf{a}_{i[t]}^3, & \text{de otra forma} \end{cases} \quad \text{(B.107)}$$

$$\forall i, \ldots, M,\ \mathbf{x}_{i[t+1]} = f(\mathbf{b}_{i[t]}) = \begin{cases} 1, & \text{si } \mathbf{b}_{i[t]} > 1 \\ -1, & \text{si } \mathbf{b}_{i[t]} > -1 \\ (\delta+1)\mathbf{b}_{i[t]} - \delta\mathbf{b}_{i[t]}^3, & \text{de otra forma} \end{cases} \quad \text{(B.108)}$$

La figura B.34 ilustra la forma de la función de salida para $\delta = 0.4$. Es similar a una función sigmoidea pero alcanza su límite ±1 en un número finito de iteraciones.

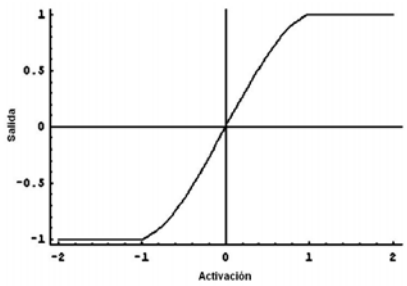



**Figura B.34** Curva de la función de salida para $\delta = 0.4$

La función de salida propuesta muestra dos mecanismos de atracción. El primero es una función de saturación rígida que limita la salida a ±1; la segunda es un mecanismo que balancea las partes positiva, $(\delta + 1)\mathbf{a}_i$, y negativa, $-\delta\mathbf{a}_i^3$. Por lo tanto, una unidad de salida permanece sin cambios si alcanza un valor de 1, -1 o si $(\delta+1)\mathbf{a}_i - R = \delta\mathbf{a}_i^3$, donde $R$ es un límite con un valor real (por ejemplo, 0.7). Este mecanismo le permite a la red exhibir un comportamiento de atractor de valores reales además, de ser un atractor bipolar.

En la regla de aprendizaje, este modelo trata de encontrar una solución a las siguientes restricciones no lineales:

$$\mathbf{X} = f(\mathbf{VY}) \quad \text{(B.109)}$$

$$\mathbf{Y} = f(\mathbf{WX}) \quad \text{(B.110)}$$

donde $f$ es la función de salida definida previamente.

Las ecuaciones para **W** y **V** son:

$$\mathbf{W} = f^{-1}\left[\left(\mathbf{Y}_{[0]}\mathbf{X}_{[0]}^T + \mathbf{Y}_{[0]}\mathbf{X}_{[t]}^T\right)\left(\mathbf{X}_{[0]} + \mathbf{X}_{[t]}\right)\times\left(\left(\mathbf{X}_{[0]} + \mathbf{X}_{[t]}\right)^T\left(\mathbf{X}_{[0]} + \mathbf{X}_{[t]}\right)^{-1}\right)\right]\times \\ \times\,\mathbf{X}_{[0]}^T\left(\mathbf{X}_{[0]}\mathbf{X}_{[0]}^T\right) \tag{B.111}$$

$$\mathbf{V} = f^{-1}\left[\left(\mathbf{X}_{[0]}\mathbf{Y}_{[0]}^T + \mathbf{X}_{[0]}\mathbf{Y}_{[0]}^T\right)\left(\mathbf{Y}_{[0]} + \mathbf{Y}_{[t]}\right)\times\left(\left(\mathbf{Y}_{[0]} + \mathbf{Y}_{[t]}\right)^T\left(\mathbf{Y}_{[0]} + \mathbf{Y}_{[t]}\right)^{-1}\right)\right]\times \\ \times\,\mathbf{Y}_{[0]}^T\left(\mathbf{Y}_{[0]}\mathbf{Y}_{[0]}^T\right) \tag{B.112}$$

Las simulaciones se realizaron con 26 asociaciones de patrones. Los patrones son imágenes de 7 x 7 pixeles, en donde a un píxel blanco se le da el valor de –1 y a un píxel negro se le da un valor de 1. Los patrones se muestran en la figura B.35

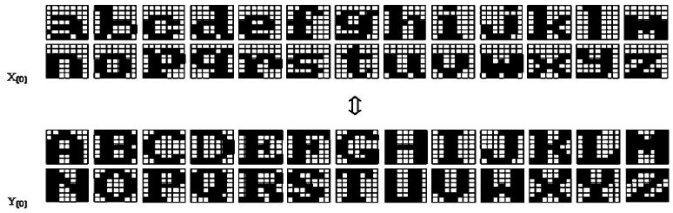


**Figura B.35** Asociaciones de los pares de patrones bipolares.

En la figura B.36 se muestra la gráfica de capacidad de almacenamiento del modelo presentado en este trabajo, comparado con otras BAM: GABAM, GBAM, ABAM, SBAM y KBAM.

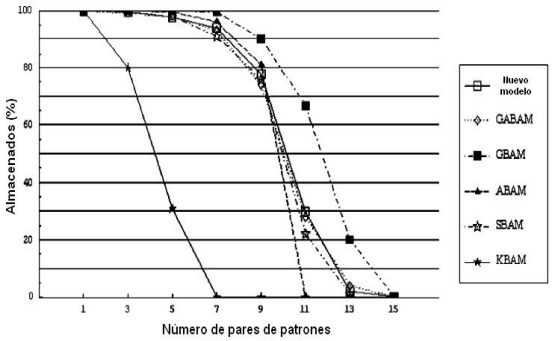



**Figura B.36** Comparación de la capacidad de almacenamiento

La gráfica de comparación muestra que la capacidad de almacenamiento de este modelo es similar a la de la SBAM, ABAM y GABAM, sin embargo es inferior a la capacidad de almacenamiento de la GBAM.

## *BAM con retardos*

En un intento por mejorar la capacidad de recuperación de las BAM, los investigadores han propuesto un nuevo camino: las BAM con retardos (*delays*) [76-84]. Estos nuevos modelos, pretenden simular el comportamiento de las neuronas y la comunicación entre ellas. Su idea se basa en que las redes neuronales biológicas presentan retardos debido al proceso de la

información, esto es, la transmisión de la información entre neuronas, así como la respuesta de éstas a una excitación, no son instantáneas. Además, en este tipo de redes neuronales, existen variedades en los tamaños y longitudes de los axones (parte de la neurona que funge como línea de transmisión).
En estos trabajos se realiza un análisis de las BAM con retardos, para encontrar las condiciones suficientes que permitan la convergencia de las memorias hacia puntos de equilibrio, los cuales deben corresponder a los patrones entrenados.

# Referencias


[1] Hassoun, M. H. 1993, *Associative Neural Memories*, Oxford University Press, New York.

[2] Kohonen, T. 1989, *Self-Organization and Associative Memory*, Springer-Verlag, Berlin.

[3] McCulloch, W. & Pitts, W. 1943, "A logical calculus of the ideas immanent in nervous activity", *Bulletin of Mathematical Biophysics*, vol. 5, pp. 115-133.

[4] Ritter, G. X. & Sussner, P. 1996, "An Introduction to Morphological Neural Networks" in *Proceedings of the 13th International Conference on Pattern Recognition*, vol. IV, Track D, pp. 709-717.

[5] Rosenblatt, F. 1958, "The Perceptron: A Probabilistic Model for Information Storage and Organization in the Brain", *Psychological Review*, vol. 65, no. 6, pp. 386-408.

[6] Widrow, B. & Lehr M. A. 1990, "30 Years of Adaptive Neural Networks: Perceptron, Madaline and Backpropagation", *Proceedings of the IEEE*, vol. 78, no. 9, pp. 1415-1441.

[7] Werbos, P. J. 1990, "Backpropagation Through Time: What It Does and How to Do It", *Proceedings of the IEEE*, vol. 78, no. 10, pp. 1550-1560.

[8] Hopfield, J.J. 1982, "Neural networks and physical systems with emergent collective computational abilities", *Proceedings of the National Academy of Sciences*, vol. 79, pp. 2554-2558.

[9] Minsky, M. & Papert, S. 1969, *Perceptrons*, MIT Press ,Cambridge.

[10] Steinbuch, K. 1961, "Die Lernmatrix", *Kybernetik*, vol. 1, no. 1, pp. 36-45.

[11] Willshaw, D., Buneman, O. & Longuet-Higgins, H. 1969, "Non-holographic associative memory", *Nature*, no. 222, pp. 960-962.

[12] Anderson, J. A. 1972, "A simple neural network generating an interactive memory", *Mathematical Biosciences*, vol. 14, pp. 197-220.

[13] Kohonen, T. 1972, "Correlation matrix memories", *IEEE Transactions on Computers*, *C-21*, vol. 4, pp. 353-359.

[14] Kosko, B. 1988, "Bidirectional associative memories", *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 18, no. 1, pp. 49-60.

[15] Yánez, C. 2002, *Memorias Asociativas basadas en Relaciones de Orden y Operadores Binarios*, Tesis de Doctorado, Centro de Investigación en Computación, México.

[16] Haines, K. & Hecht-Nielsen, R., 1988, "A BAM With Increased Information Storage Capacity", *IEEE International Conference on Neural Networks*, vol. 1, pp. 181-190

[17] Leung, C.S. & Cheung, K.F., 1991, "Householder encoding for discrete bidirectional associative memory**",** *IEEE International Joint Conference on Neural Networks*, Vol. 1, pp. 237-241.

[18] Shen, D. & Cruz, J.B., Jr., 2005, "Encoding strategy for maximum noise tolerance bidirectional associative memory", *IEEE Transactions on Neural Networks*, Vol. 16, Issue 2, pp. 293-300

[19] Wang, Y.-F., Cruz J.B., Jr. & Mulligan, Jr., 1991, "Guaranteed recall of all training pairs for bidirectional associative memory", *IEEE Transactions on Neural Networks*, Vol. 1, Issue 6, pp. 559-567

[20] Simpson, P. K. 1990, *Artificial Neural Systems*, Pergamon Press , New York.

[21] Steinbuch, K. & Frank, H. 1961, "Nichtdigitale Lernmatrizen als Perzeptoren", *Kybernetik*, vol. 1, no.3, pp. 117-124.

[22] Papadomanolakis, K., Kakarountas, A., Sklavos, N. & Goutis C. E., 2002, A Fast Johnson-Mobius Encoding Scheme for Fault Secure Binary Counters, *Proceedings of Design, Autamtions and Test in Europe*, (DATE '02), pp. 1-7.

[23] Anderson, J. A. & Rosenfeld, E. 1990, *Neurocomputing: Fundations of Research*, MIT Press ,Cambridge.

[24] Abu-Mostafa, Y. & St. Jacques, J. 1985, "Information capacity of the Hopfield model", *IEEE Transactions on Information Theory*, IT-31, no. 4, pp. 461-464.

[25] Ritter, G. X., Sussner, P. & Diaz-de-Leon, J. L. 1998, "Morphological associative memories", *IEEE Transactions on Neural Networks*, vol. 9, pp. 281-293.

[26] Sossa, H., Barrón, R. & Vázquez, R. 2004, "New Associative Memories to Recall Real-Valued Patterns", *CIARP*, LNCS 3287, pp. 195-202.

[27] Austin, J. 1987, "ADAM: A Distributed Associative Memory for Scene Analysis", In *Proceedings of First International Conference on Neural Networks*, pp. 285-295.

[28] Wang, Y.-F., Cruz J.B., Jr. & Mulligan, Jr., 1989, "An enhanced bidirectional associative memory", *IJCNN International Joint Conference on Neural Networks*, Vol. 1, pp. 105-110

[29] Tai, H.-M., Wu, C.-H. & Jong, T.-L., 1989, "High-order bidirectional associative memory", *Electronics Letters*, Vol. 25, Issue 21, pp. 1424-1425

[30] Jeng, Y.-J.; Yeh, C.-C. & Chiveh, T.D., 1990, "Exponential bidirectional associative memories**",** *Electronics Letters*, Vol. 26, Issue 11, pp. 717-718

[31] Wang, Y.-F., Cruz J.B., Jr. & Mulligan, Jr., 1990, "Two coding strategies for bidirectional associative memory", *IEEE Transactions on Neural Networks*, Vol. 1, Issue 1, pp. 81-92

[32] Wang, Y.-F., Cruz J.B., Jr. & Mulligan, Jr., 1990, "On multiple training for bidirectional associative memory", *IEEE Transactions on Neural Networks*, Vol. 1, Issue 3, pp. 275-276

[33] Simpson, P.K., 1990, "Higher-Ordered and Intraconnected Bidirectional Associative Memories", *IEEE Transactions on Systems, Man, and Cybernetics*, Vol. 20, No. 3, pp. 637-653

[34] Srinivasan, V. & Chia, C.S., 1991, "Improving bidirectional associative memory performance by unlearning", *IEEE International Joint Conference on Neural Networks*, Vol. 3, pp. 2472-2477

[35] Wang, W.-J. & Lee, D.-L., 1992, "Modified exponential bidirectional associative memories", *Electronics Letters*, Vol. 28, Issue 9, pp. 888-890

[36] Jeng, Y.-J., Yeh, C.-C. & Chiueh, T.-D., 1992, "Generalised stable bidirectional associative memory**",** *Electronics Letters*, Vol. 28, Issue 15, pp. 1396-1398

[37] Yu, H. & Wang, Z., 1992, "Bidirectional associative memories on the matching criterion of weighted Hamming distance", *Proceedings of the 31st IEEE Conference on Decision and Control*, Vol. 4, pp. 3481-3484

[38] Lee, D.-L. & Wang, W.-J., 1993, "Improvement of bidirectional associative memories by using correlation significance", *Electronics Letters*, Vol. 29, Issue 8, pp. 688-690

[39] Perfetti, R., 1993, "Optimal gradient descent learning for bidirectional associative memories", *Electronics Letters*, Vol. 29, Issue 17, pp. 1556-1557

[40] Leung, C.S., 1993, "Encoding method for bidirectional associative memory using projection on convex sets", *IEEE Transactions on Neural Networks*, Vol. 4, Issue 5, pp. 879-881

[41] Leung, C.S., 1993, Robust learning rule for bidirectional associative memory", *Proceedings of IJCNN International Joint Conference onNeural Networks*, Vol. 3, pp. 2686-2689

[42] Wang, W. & Lee, D. 1993, "A modified bidirectional decoding strategy based on the BAM structure", *IEEE Trans. Neural Network*, col. 4, no. 4, pp. 710-717

[43] Khorasani, K., Cuffaro, A. & Grigoriu, T., 1994, “A new learning algorithm for bidirectional associative memory neural networks”, *IEEE International Conference on Neural Networks*, Vol. 2, pp. 1115-1120

[44] Wang, C.-C. & Tsai, C.-R., 1994, “An analysis of practical capacity of exponential bidirectional associative memory”, *IEEE International Conference on Neural Networks*, Vol. 2, pp. 1074-1079

[45] Xu, Z.-B., Leung, Y. & He, X.-W., 1994, “Asymmetric bidirectional associative memories”, *IEEE Transactions on Systems, Man and Cybernetics*, Vol. 24, Issue 10, pp. 1558-1564

[46] Hattori, M., Hagiwara, M. & Nakagawa, M., 1994, “New results of Quick Learning for Bidirectional Associative Memory having high capacity”, *IEEE International Conference on Neural Networks*, Vol. 2, pp. 1080-1085

[47] Oh, H. & Kothari, S. 1992, ”A Pseudo-Relaxation Learning Algorithm for Bidirectional Associative Memory”, *IJCNN*, vol. 2, pp. 208-213

[48] Leung, C.S., 1994, “Optimum learning for bidirectional associative memory in the sense of capacity”, *IEEE Transactions on Systems, Man and Cybernetics*, Vol. 24, Issue 5, pp. 791-796

[49] Hu, G.-Q., Kwong, C.-P. & Xu, Z.-B., 1994, “Two iterative encoding schemes for bidirectional associative memory”, *Proceedings of ISSIPNN International Symposium on Speech, Image Processing and Neural Networks*, Vol. 1, pp. 93-96

[50] Wang, Z.-O., 1996, “A bidirectional associative memory based on optimal linear associative memory”, *IEEE Transactions on Computers*, Vol. 45, Issue 10, pp. 1171-1179

[51] Sarkar, D., 1996, “A three-stage architecture for bidirectional associative memory”, *IEEE International Conference on Neural Networks*, Vol. 1, pp. 531/4-531/9

[52] Osana, Y., Hattori, M. & Hagiwara, M., 1996, “Chaotic bidirectional associative memory”, *IEEE International Conference on Neural Networks*, Vol. 2, pp. 816-821

[53] Chen, S., Gao, H. & Yan, W., 1997, “Improved exponential bidirectional associative memory”, *Electronics Letters*, Vol. 33, Issue 3, pp. 223-224

[54] Haryono, Sadananda, R. & Phien, H.N., 1997, “Orthogonal schemes for bidirectional associative memories”, *IEEE Transactions on Systems, Man and Cybernetics*, Part B, Vol. 27, Issue 3, pp. 543-551

[55] Araújo, A.F.R. & Haga, G.M., 1997, “Two simple strategies to improve bidirectional associative memory's performance: unlearning and delta rule”, *International Conference on Neural Networks*, Vol. 2, pp. 1178-1182

[56] Widrow, B. & Hoff, M. 1960, *Adaptive switching circuits*, IRE WESCON Convention Record, New York

[57] Ritter, G.X., Diaz-deLeon, J.L. & Sussner, P., 1999, "Morphological bidirectional associative memories", *Neural Networks*, Vol. 12, pp. 851-867

[58] Wu,Y. & Pados, D.A., 2000, "A feedforward bidirectional associative memory", *IEEE Transactions on Neural Networks*, Vol. 11, Issue 4, pp. 859-866

[59] Lenze, B., 2001, "Improving Leung's bidirectional learning rule for associative memories", *IEEE Transactions on Neural Networks*, Vol. 12, Issue 5, pp. 1222-1226

[60] Eom, T.-D., Oh, S.-K. & Lee, J.-J., 2001, "Guaranteed recall of all training pairs for exponential bidirectional associative memory", *Electronics Letters*, Vol. 37, Issue 3, pp. 153-154

[61] Oh, H. & Kothari, S. 1994, "Adaptation of the relaxation method for learning in bidirectional associative memory", *IEEE Trans. Neural Network*, vol. 5, no. 4, pp. 576-583.

[62] Shi, H., Zhao, Y. & Zhuang, X. 1998, "A general model for bidirectional associative memories", *IEEE Trans. Sys. Man. Cybern. B.*, vol. 28, pp. 511-519

[63] Lee, D.-L. & Chuang, T.C., 2004, "Design of General Bidirectional Associative Memories with Improved Recall Capability", *International Journal of Neural Systems*, Vol. 14, No. 5, pp. 325-328

[64] Zheng, G., Givigi, S.N. & Zheng, W., 2005, "A New Strategy for Designing Bidirectional Associative Memories", *Lecture Notes in Computer Science*, Springer-Verlag, Vol. 3496, pp. 398-403

[65] Chartier, S. & Boukadoum, M., 2006, "A Bidirectional Heteroassociative Memory for Binary and Grey-Level Patterns", *IEEE Transactions on Neural Networks*, Vol. 17, No. 2, pp. 385-396.

[66] Rosen, K., 1999, *Discrete Mathematics and Its Applications*, McGraw-Hill, Estados Unidos.

[67] http://bias.csr.unibo.it/fvc2000/download.asp

[68] Duda, R. O. & Hart, P. E., 1973, Pattern Classification and Scene Analisys, New York: John Wiley & Sons.

[69] Friedman, M. & Kandel, A., 2000, Introduction to Pattern Recognition (Statistical, Structural, Neural and Fuzzy logiz Approaches), Singapore, World Scientific.

[70] López, F., Álvarez, I., Acevedo, M., García, C. & Macías, M., 2003, *Neural Proccessing Letters*, Vol. 17, pp. 137-148.

[71] Wang, C. C. & Tsai C. R., 1998, Data Compression by the Recursive Algorithm of Exponential Associative Memory, *IEEE Transactions on Systems, Man, and Cybernetics-Part B: Cynernetics*, Vol. 28, No. 2, pp. 125-134.

[72] Tryon, W., 1999, A Bidirectional Associative Memory Explanation of Posttraumatic Stress Disorder, *Clinical Phychology Review*, Vol. 19, No. 7, pp. 789-817.

[73] Bělohlávec, R., 2000, Representation of Concept Lattices by Bidirectional Associative Memories, Neural Computation, Vol. 12, pp. 2279-2290.

[74] Rajapakse, R. & Denham, M., 2005, Fast Acces to Concepts in Concept Lattices via Bidirectional Associative Memories, *Neural Computation*, Vol. 17, pp. 2291-2300.

[75] Domínguez, E. & Muñoz, J., 2004, Bidirectional Neural Network for Clustering Problems, *Springer-Verlag* IBERAMIA, LNAI 3315 pp. 788-798

[76] Gopalsamy, K. & He, X-Z., 1994, "Delay-Independent Stability in Bidirectional Associatuve Memory Networks", *IEEE Transactions on Neural Networks*, vol. 5, no. 6, pp. 998-1002

[77] Rao, S. & Phaneendra, R., 1999, "Global dynamics of bidirectional associative memory neural networks involving transmission delays and dead zones", *Neural Networks*, vol. 12, pp. 455-465

[78] Cao, J. & Wang, L., 2000, "Periodic oscillatory solution of bidirectional associative memory neural networks with delays", *Physical Review E*, vol. 61, no. 2, pp. 1825-1828

[79] Zhang, J. & Yang, J., 2001, "Global stability analysis of bidirectional associative memory neural networks with delays", *Int. J. Circ. Theor. Appl.*, vol. 29, pp. 185-196

[80] Zhao, H., 2002, "Global stability analysis of bidirectional associative memory neural networks with distributed delays", *Physics Letters A*, vol. 297, pp. 182-190

[81] Feng, C. & Plamondon, R., 2003, "Stability Analysis of Bidirectional Associative Memory Neural Networks With Time Delays", *IEEE Transactions on Neural Networks*, vol. 14, no. 6 pp. 1560-1565

[82] Liang, J. & Cao, J., 2004, "Exponential stability of continuous-time and discrete time bidirectional associative memory neural networks with delays", Chaos, Solitons and Fractals, vol. 22, pp. 773-785

[83] Arik, S., 2005, "Global asymptotic stability analysis of bidirectional associative memory neural networks with time delays", *IEEE Transactions on Neural Networks*, vol. 16, no. 3 pp. 580-585

[84] Park, J., 2006, "Robust stability of bidirectional associative memory neural networks with time delays", *Physics Letters A*, vol. 349, pp. 494-499