4

REDES DE INTERCONEXIÓN

4.1. Introducción

Evidentemente, un sistema con varios procesadores precisa de unos medios de comunicación entre dichos procesadores. Esta función de comunicación puede realizarse básicamente de dos formas:

- Compartiendo algún recurso, tradicionalmente la memoria. Será a través de accesos a ese recurso, por parte de los diversos procesadores, como será posible el intercambio de información. En este caso se dirá que se ha establecido la comunicación por **memoria compartida**.

- A través del envío de copias de las informaciones desde cada uno de los procesadores hacia los demás elementos de proceso con los que se quiera comunicar. En este segundo caso hablaremos de comunicación por **paso de mensajes**

Estos dos métodos concuerdan con las dos filosofías diferentes analizadas en el apartado 1.4.2, respecto a la distribución o compartición de la memoria. El primer método se utiliza en los sistemas multiprocesadores, mientras que el segundo se utiliza en los sistemas multicomputadores. Debe quedar claro, no obstante, que estas formas de trabajo a bajo nivel pueden quedar enmascaradas por el software; esto significa que el programador puede emplear un modelo de programación (paso de mensajes o variables compartidas) e internamente utilizarse el otro.

En ambos casos será necesaria una vía de comunicación física, bien para unir físicamente la memoria compartida a todos los procesadores, o bien para poder enviar mensajes de un elemento de proceso a otro.

4.2. Rendimiento de los sistemas de comunicación entre procesadores

En este párrafo analizaremos la modificación del rendimiento de los sistemas por el hecho de añadir más elementos de proceso. Habrá que tener en cuenta los beneficios del proceso paralelo, pero también se deberá contabilizar la sobrecarga que suponen las comunicaciones. Para modelar matemáticamente estos rendimientos plantearemos diferentes perfiles de comunicación. Esto dará lugar a varios modelos que analizaremos a continuación. Todos estos modelos se deben al trabajo de Indurkhya, Stone y Xi-Cheng (1986).

4.2.1. Modelo básico: dos procesadores con comunicación total entre los procesos

En este modelo, consideraremos un sistema con dos procesadores en el que se ejecutan m tareas. El modelo parte de las siguientes hipótesis:

- El tiempo de cálculo puro de cada tarea es R unidades de tiempo.
- Cada tarea se comunica con cada una de las demás con un coste en tiempo adicional de C unidades, si las tareas que se comunican se ejecutan en procesadores distintos. No hay coste adicional de comunicaciones apreciable si las tareas se ejecutan en el mismo procesador.

Según estas hipótesis, supongamos que, de las m tareas, k se ejecutan en un procesador y $m-k$ en el otro. En estas condiciones, el tiempo de ejecución de las tareas repartidas entre ambos procesadores será:

$$T = R\,\text{máx}(k, m-k) + C(m-k)k \qquad [4.1]$$

Los dos términos de esta ecuación representan, respectivamente, al tiempo de cálculo y al tiempo de comunicaciones. Sería interesante calcular en qué condiciones el tiempo total de ejecución, dado por la expresión 4.1, se hace mínimo. En la figura 4.1 puede verse la representación gráfica de la citada expresión, tomando R como unidad (el tiempo que tarda en ejecutarse cada tarea en un solo procesador), para 100 tareas y diferentes valores de la relación R/C. Como se observa de forma evidente, ambos términos de 4.1 tienen un extremo relativo en $k = m/2$, aunque de diferente clase; asimismo, ambos términos tienen sendos extremos absolutos en $k = 0$ y $k = m$, pero también de diferente clase. Como puede verse en la citada figura, para $R/C = 10$ es contraproducente repartir las tareas entre ambos procesadores, ya que el tiempo total de ejecución obtenido es mayor que cuando se ejecutan todas en uno solo; no ocurre lo mismo para $R/C = 100$, en que el tiempo de ejecución con la carga repartida es menor. Calcularemos ahora el valor de R/C para el que el tiempo total de ejecución es el mismo en ambos casos (con la carga repartida entre los procesadores y con las tareas concentradas en un solo procesador). A partir de ese punto, el mínimo central será el mínimo absoluto. Este valor lo obtendremos igualando el valor de la expresión 4.1 en $k = 0$ y $k = m/2$, con lo que se obtiene:

$$T(0) = Rm \qquad T\left(\frac{m}{2}\right) = R\frac{m}{2} + C\frac{m^2}{4}$$

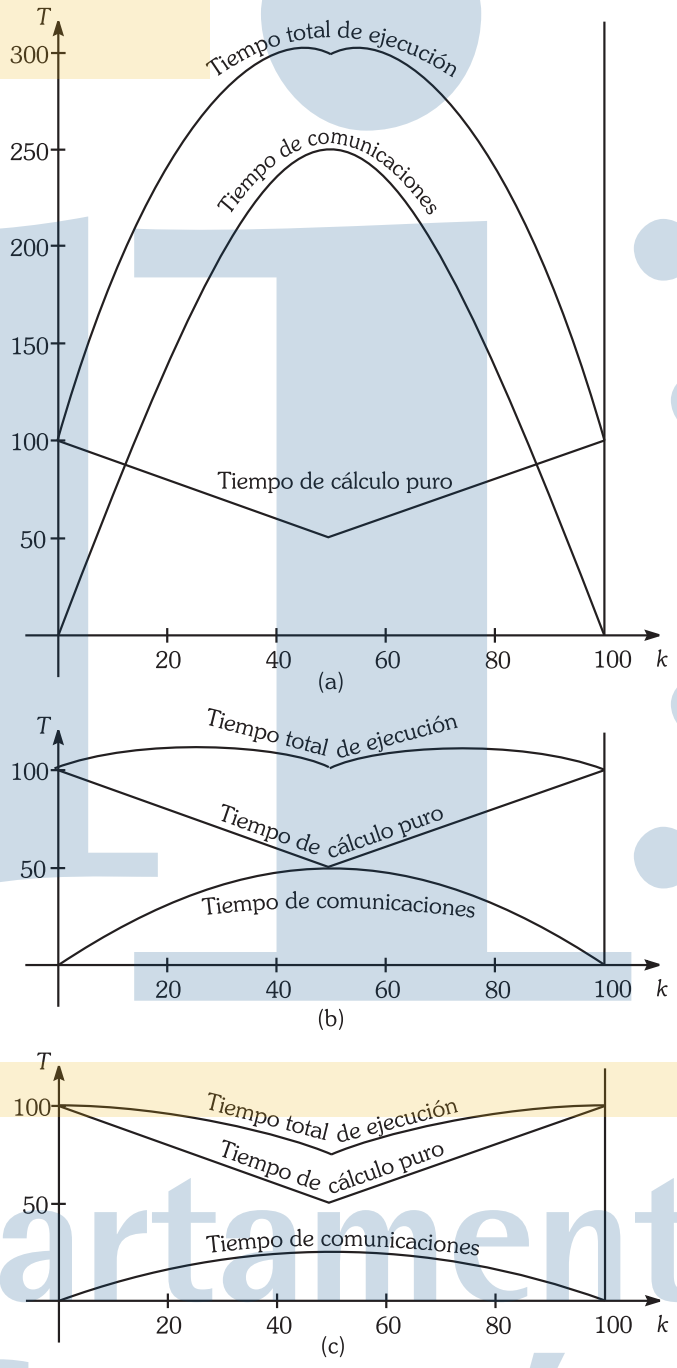


Fig. 4.1. Representación del tiempo de ejecución en función del reparto de 100 tareas en dos procesadores para diferentes valores de la relación R/C: (a) $R/C = 10$, (b) $R/C = 50$ y (c) $R/C = 100$ (modelo cuadrático para el tiempo de comunicaciones).

$$Rm = R\frac{m}{2} + C\frac{m^2}{4} \quad \Longrightarrow \quad \frac{R}{C} = \frac{m}{2}$$

El significado de este resultado es muy claro: para $R/C < m/2$, el mínimo se encuentra en $k = 0$ y en $k = m$, es decir, es mejor dejar toda la carga en un solo procesador. Esto significa que no es rentable utilizar los dos procesadores, ya que el tiempo perdido por las comunicaciones es muy alto. Por el contrario, si $R/C > m/2$ el mínimo se encuentra en $k = m/2$ puede ser mejor utilizar los dos procesadores siempre que se equilibre la carga entre ambos.

Hay que tener en cuenta que el rendimiento obtenido para $R/C = m/2$ es bastante pobre, ya que se están utilizando dos procesadores para obtener el mismo tiempo de ejecución que con uno solo (rendimiento del 50 %). Si la relación R/C va aumentando por encima de $m/2$, el rendimiento comenzará a incrementarse.

4.2.2. Modelo extendido: N procesadores con comunicación total entre los procesos

Consideraremos ahora el mismo modelo pero con N procesadores en vez de 2. En este caso, llamaremos k_i al número de procesos que se ejecutan en el procesador i. Generalizando la ecuación 4.1 a N procesadores, tendremos que:

$$\begin{aligned} T &= R\,\text{máx}(k_i)_{i=1,..,N} + \frac{C}{2}\sum_{i=1}^{N} k_i(m - k_i) = \\ &= R\,\text{máx}(k_i)_{i=1,..,N} + \frac{C}{2}\left(m\sum_{i=1}^{N} k_i - \sum_{i=1}^{N} k_i^2\right) = \\ &= R\,\text{máx}(k_i)_{i=1,..,N} + \frac{C}{2}\left(m^2 - \sum_{i=1}^{N} k_i^2\right) \end{aligned} \qquad [4.2]$$

Puede parecer extraño que en el segundo término aparezca $C/2$ en lugar de solamente C, esto se debe a que en la suma se han contado los términos dos veces: una en el procesador origen de la comunicación, y otra en el procesador destino.

Parece lógico pensar, por analogía con el modelo anterior, que los mínimos de la expresión 4.2 se encontrarán, bien cuando la carga esté repartida uniformemente entre los N procesadores, o bien cuando toda esté concentrada en uno de ellos. Similarmente al modelo anterior, existirá un momento a partir del que uno de los mínimos empezará a prevalecer frente al otro. Ese momento ocurrirá cuando ambos mínimos coincidan, es decir:

$$\overbrace{Rm + \frac{C}{2}(m^2 - m^2)}^{\text{(1 procesador)}} \quad = \quad \overbrace{R\frac{m}{N} + \frac{C}{2}\left(m^2 - \sum_{i=1}^{N}\left(\frac{m}{N}\right)^2\right)}^{N \text{ procesadores con la carga repartida}}$$

A partir de ahí haremos las operaciones necesarias para calcular la relación R/C:

$$Rm = R\frac{m}{N} + \frac{C}{2}\left(m^2 - \frac{m^2}{N}\right) = R\frac{m}{N} + \frac{Cm^2}{2}\left(1 - \frac{1}{N}\right)$$

$$Rm\left(1 - \frac{1}{N}\right) = \frac{Cm^2}{2}\left(1 - \frac{1}{N}\right) \Longrightarrow \quad Rm = \frac{Cm^2}{2} \quad \Longrightarrow \quad \frac{R}{C} = \frac{m}{2} \qquad [4.3]$$

Como puede apreciarse, este resultado es exactamente el mismo que el obtenido para 2 procesadores: Si $R/C > m/2$ puede ser ventajoso utilizar varios procesadores; si $R/C < m/2$, el tiempo empleado en las comunicaciones no compensa el uso de varios procesadores. Sin embargo, si analizamos el rendimiento, veremos que para $R/C = m/2$, en este caso es peor que con dos procesadores, ya que el rendimiento será $1/N$, es decir, se están utilizando N procesadores para conseguir el mismo tiempo de ejecución que con uno.

4.2.3. Modelo lineal en el tiempo de comunicaciones

El modelo anterior es bastante pesimista, ya que se supone que el perfil de comunicaciones de los procesos es muy amplio, hasta el punto de que cada uno de los procesos se comunica con todos los demás. Normalmente, la necesidad de comunicaciones no es tan alta. El modelo anterior suponía un perfil cuadrático en las comunicaciones (ecuación 4.2). Ahora supondremos que las comunicaciones tienen sólo un peso lineal en el tiempo total de cálculo. En este modelo, el peso de las comunicaciones será proporcional al número de procesadores. Por tanto, el tiempo total de ejecución vendrá dado por:

$$T = R\,\text{máx}(k_i) + CN \qquad [4.4]$$

En esta ecuación, el primer término depende de la asignación de tareas a los procesadores, no así el segundo; en él ya se ha supuesto que los procesos están repartidos, y que la carga de la comunicación sólo depende del número total de procesadores. Como era de esperar, será en esta situación de procesos repartidos en la que se conseguirá el mejor rendimiento, ya que así se consigue que el primer término adquiera su menor valor posible, que es Rm/N. Esto hace que la ecuación 4.4, en este caso, se transforme en

$$T = \frac{Rm}{N} + CN \qquad [4.5]$$

También hay que tener en cuenta que, si se incrementa el valor de N, también crece el segundo termino y llegará un momento en que ese término crezca más deprisa de lo que decrece el otro, haciendo decrecer el rendimiento. Para calcular el número de procesadores para el que eso ocurre, calcularemos la derivada de la expresión 4.5 respecto a N y la igualaremos a 0, con lo que obtendremos:

$$\frac{dT}{dN} = -\frac{Rm}{N^2} + C, \quad -\frac{Rm}{N^2} + C = 0 \quad \Longrightarrow \quad N = \sqrt{\frac{Rm}{C}} \qquad [4.6]$$

Si, para este valor, calculáramos la derivada segunda de T respecto a N, obtendríamos que es positiva lo que confirma que ese valor de N obedece a un mínimo.

Este resultado puede resultar bastante paradójico porque parece lógico pensar que m tareas se ejecutarán mejor en m procesadores diferentes. Sin embargo, según lo anterior, esto no es así ya que el número óptimo de procesadores crece con la raíz cuadrada de m. Las razones de este resultado estriban en el coste adicional debido a las comunicaciones.

4.2.4. Modelo óptimo

En este apartado estudiaremos el caso óptimo, según el cual todas las comunicaciones pueden superponerse con el tiempo de cálculo. Esta situación sería posible, al menos en parte, si el sistema dispusiera de elementos específicos de comunicación totalmente independientes del propio procesador (por ejemplo, procesadores de entrada/salida). Utilizaremos para nuestros cálculos de este caso óptimo, el modelo cuadrático para el tiempo de comunicaciones descrito en el apartado 4.2.2. Según estas premisas, el tiempo total de ejecución de las m tareas vendría dado por:

$$T = \text{máx}\left(R\,\text{máx}(k_i)_{i=1,..,N}, \frac{C}{2}\sum_{i=1}^{N} k_i(m-k_i)\right) \qquad [4.7]$$

Para el caso de dos procesadores, tomaremos el máximo de los dos términos de la ecuación 4.1, con lo que las curvas de la figura 4.1 se transformarán en las mostradas en la figura 4.2 (en la que el tiempo total de ejecución se ha representado mediante trazo grueso). Si hay corte entre las representaciones de ambos términos de la ecuación 4.1 (figura 4.2(a)), el mínimo tiempo de ejecución se encontrará en ese corte y podrá calcularse igualando ambos términos para obtener el valor más conveniente de k. De esta forma obtenemos:

$$R(m-k) = C(m-k)k \quad \Longrightarrow \quad k = \frac{R}{C}$$

Teniendo en cuenta la simetría del problema, los valores de k, están restringidos al intervalo $[0, m/2]$, para que el problema no se repita intercambiando los procesadores. Esto implica que, en función del valor de R/C, el mínimo recorre también este intervalo. El tiempo de ejecución en ese mínimo será:

$$T = R(m-k_{min}) = R\left(m - \frac{R}{C}\right)$$

y la ganancia de velocidad será:

$$S = \frac{t(1)}{t(N)} = \frac{Rm}{R(m-\frac{R}{C})} = \frac{1}{1-\frac{R}{Cm}}$$

En el intervalo de variación de R/C, es decir $[0, m/2]$, esta ganancia de velocidad variará entre 1 (para $R/C = 0$) y 2 (para $R/C = m/2$). Este último valor corresponde a los casos mostrados en la figura 4.2(b), en que la curva que representa el segundo término de 4.1 queda por debajo de la que representa al primero.

Para N procesadores, sabemos que el costo mínimo de cálculo se produce cuando la carga está equilibrada entre todos los procesadores. Para este caso, con nuestro modelo óptimo, el tiempo total de ejecución, dado genéricamente por la ecuación 4.7, será:

$$T = \text{máx}\left(\frac{Rm}{N}, \frac{Cm^2}{2}\left(1-\frac{1}{N}\right)\right)$$

El mejor resultado se producirá cuando:

$$\frac{Rm}{N} \geq \frac{Cm^2}{2}\left(1-\frac{1}{N}\right)$$

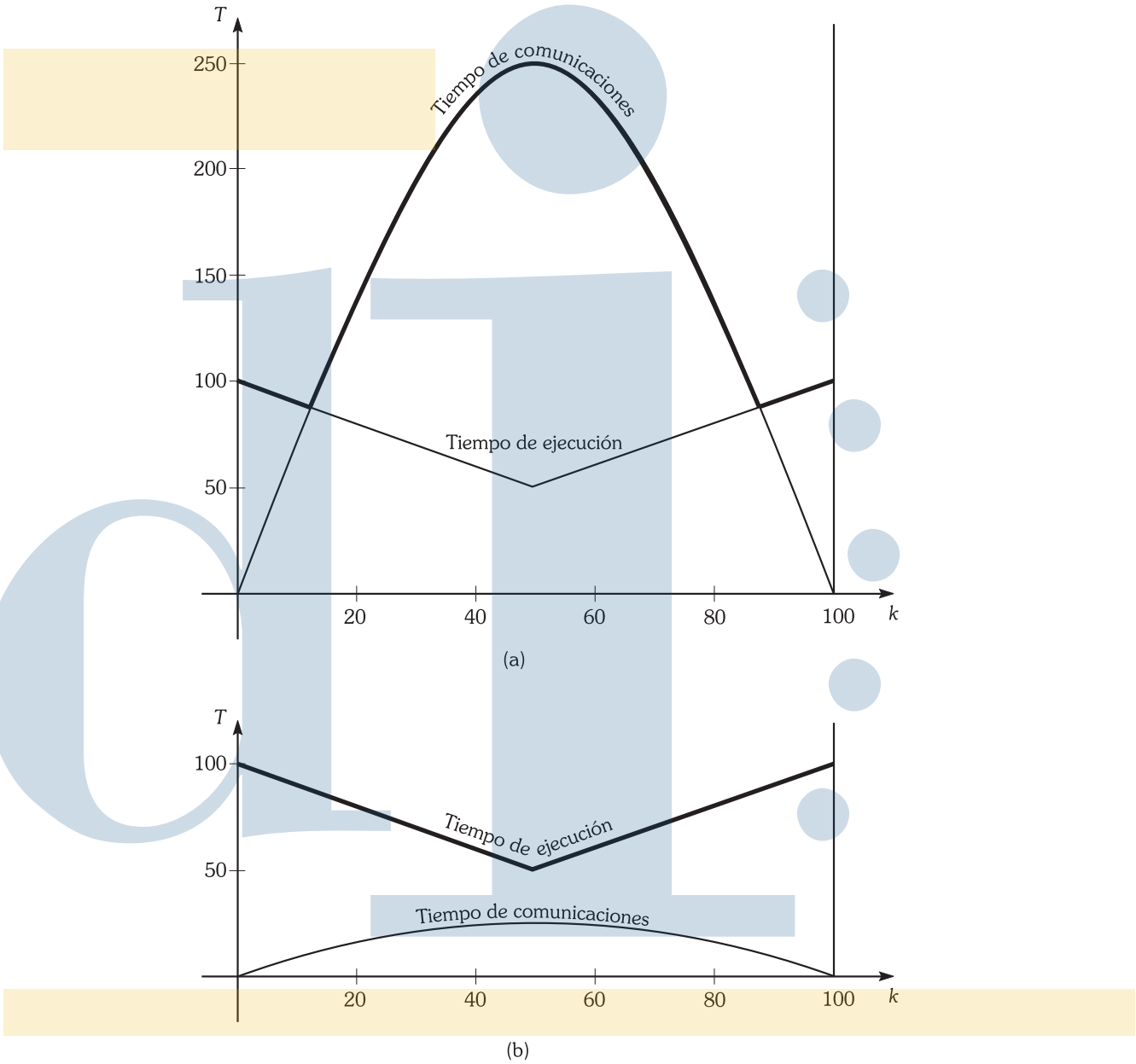


Fig. 4.2. Representación del tiempo total de ejecución (trazo grueso) en función del reparto de tareas en dos procesadores para diferentes valores de la relación R/C: (a) $R/C = 10$ y (b) $R/C = 100$ (modelo óptimo).

Simplificando y operando sobre esta expresión para despejar N, tenemos:

$$\frac{R}{N} \geq \frac{Cm}{2}\left(1 - \frac{1}{N}\right) \Longrightarrow \frac{R}{N} \geq \frac{Cm}{2}\left(\frac{N-1}{N}\right) \Longrightarrow$$

$$R \geq \frac{Cm}{2}(N-1) \Longrightarrow \frac{2R}{Cm} \geq N - 1 \Longrightarrow \frac{2R}{Cm} + 1 \geq N$$

Es decir, el número de procesadores deberá cumplir la condición:

$$N \leq \frac{2R}{Cm} + 1$$

Como puede verse, con este modelo, la cota superior para el número de procesadores óptimo es inversamente proporcional al número de tareas. Este hecho, que puede parecer bastante sorprendente, es debido al modelo elegido para el tiempo de comunicaciones. Si la carga de las comunicaciones fuera inferior el resultado sería totalmente diferente (ver ejercicio 4.3).

4.2.5. Conclusiones

Como conclusiones a todos estos estudios, podemos extraer las siguientes:

1. Las arquitecturas paralelas producen un costo adicional (en tiempo) debido a las comunicaciones entre los procesadores. Este tipo de costo no se produce en los procesadores escalares o vectoriales (es decir, en los procesadores con flujo único de instrucciones). Bajo el concepto ”costo de comunicaciones” se incluyen tanto el tiempo necesario para intercambiar datos entre los procesadores, como los tiempos gastados en esperas por recursos compartidos, sincronizaciones, etc.

2. Aunque el tiempo de cálculo de los programas disminuye con el número de procesadores, no ocurre lo mismo con el tiempo necesario para las comunicaciones. Esto implica que, por aumentar el número de procesadores, no siempre aumenta la ganancia de velocidad.

3. La relación R/C mide la sobrecarga de tiempo debido a las comunicaciones respecto al tiempo de cálculo puro (en forma inversa). Evidentemente, esta relación depende de la configuración del sistema (memoria compartida o distribuida, calidad de la red de comunicación, etc.) y también del algoritmo empleado. Cuanto mayor sea la citada relación, más eficiencia se conseguirá con un sistema paralelo.

4.3. Características de las redes de interconexión

Antes de empezar a definir las características de las redes de interconexión diremos que se denomina **nodo** a cualquiera de los dispositivos que se quiera conectar a la red, tales como elementos de proceso, módulos de memoria, procesadores de entrada/salida, etc.

Definiremos a continuación algunos conceptos que nos ayudarán a caracterizar y especificar una red de interconexión:

Grado de los nodos: se llama **grado de un nodo** (d) al número de enlaces que tiene con otros nodos. En el caso de enlaces unidireccionales se puede hablar de **grado de entrada** y de **grado de salida**; en este caso se considera el grado del nodo a la suma de ambos. Es conveniente que en una red el grado sea igual para todos los nodos; en ese caso se dice que la red es **regular**. La regularidad de las redes es conveniente de cara a la modularidad y escalabilidad. Por otra parte, el grado de un nodo incide de forma clara en su coste, ya que indica el número de puertos de entrada/salida que necesita.

Diámetro de una red (D): se denomina así al máximo camino más corto entre dos nodos medido por el número de enlaces recorridos. Un diámetro menor indica mayor habilidad de comunicación en la red. Evidentemente, debe procurarse que el diámetro de las redes de intercomunicación sea lo más pequeño posible.

Ancho de bisección: cuando una determinada red se divide en dos partes iguales, el mínimo número de enlaces cortados por la división se llama ancho de bisección (b). Si cada enlace tiene w bits entonces el ancho de bisección medido en bits (B) será $B = bw$. En general, B será una cota superior de los anchos en otros cortes dentro de la red. Una medida de la capacidad de comunicación de la red es el producto del ancho de bisección por el ancho de banda de cada enlace, a este producto se le denomina **ancho de banda de bisección**.

Latencia de una red: este parámetro se refiere al retraso máximo producido por la comunicación de un mensaje elemental a través de la red. Esto está relacionado con los tiempos de espera producidos por vías de comunicación ocupadas. En algunas ocasiones, a este parámetro también se le denomina genéricamente **contención**.

Productividad: se llama productividad de una red al número total de paquetes de información (mensajes) que la red puede transportar por unidad de tiempo. A este respecto, se llama **punto caliente** (***hot spot***) de una red a un enlace que concentra una parte desproporcionadamente grande del tráfico de total de la red.

Escalabilidad: esta propiedad se refiere a la facilidad con que la red puede expandirse manteniendo sus prestaciones sin aumentar desproporcionadamente el coste.

Simetría: diremos que una red de interconexión es simétrica si su topología tiene el mismo aspecto vista desde cualquier nodo; con palabras más coloquiales podríamos decir que una red es simétrica si todos sus nodos son *intercambiables*; esto implica que una red simétrica tiene que ser regular.

Conectividad: se dice que una red tiene **acceso total** (o que está **totalmente conectada**) si permite la conexión directa entre dos nodos cualesquiera de la red. Esto implica que existan enlaces que unan todas las parejas posibles de nodos sin pasar por otros nodos. En caso contrario, se dice que la red está **parcialmente conectada**. Por otra parte, se dice que una red es **bloqueante** si no es posible conectar, en algún caso, todas las posibles parejas de nodos que se puedan formar; en caso contrario se dice que la red es **no bloqueante**. Una red se dice que es **reordenable**, o **reacondicionable**, si puede efectuar la conexión entre cualquier pareja de nodos, aunque para ello sea necesario cambiar algunas de las conexiones para conservar los enlaces existentes entre los nodos.

4.4. Elementos de conmutación (*switches*)

Muchas redes de interconexión necesitan elementos específicos para comunicar sus nodos de forma selectiva. El más habitual de estos elementos es el conmutador o *switch*.

Un **módulo conmutador** (o *switch*) $p \times q$ es un circuito con p entradas y q salidas (figura 4.3). El circuito puede conectar cualquiera de las entradas con una o varias de sus salidas.

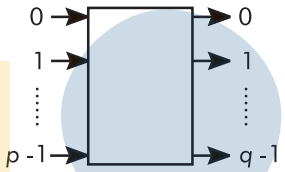


Fig. 4.3. Módulo conmutador $p \times q$.

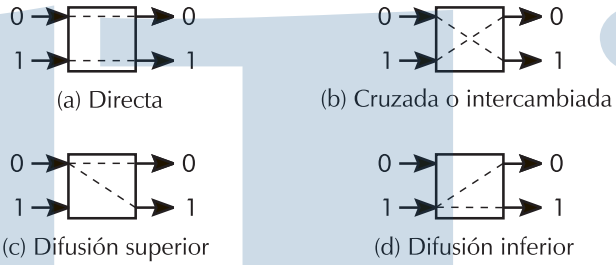


Fig. 4.4. Posiciones permitidas para un módulo conmutador 2×2.

En la práctica, lo más habitual es que $p = q = 2^r (r > 0)$ y no está permitida la conexión de una de las salidas con varias entradas, ya que esto produciría conflictos en la salida; por ello un conmutador de 2×2 podrá adquirir las 4 posiciones mostradas en la figura 4.4 denominadas **directa** (***straight***), **cruzada** o **intercambiada** (***crossover*** o ***swap***), **difusión superior** (***upper broadcast***) y **difusión inferior** (***lower broadcast***).

Si sólo se permite la conexión de cada entrada con una sola de las salidas el circuito se llama **conmutador de líneas cruzadas** (***crossbar switch***). El más habitual de estos conmutadores es el binario en que $p = q = 2$ que sólo puede tener dos de las posiciones de la figura 4.4: directa y cruzada.

4.5. Permutaciones y funciones de intercambio

Como es conocido, se llaman **permutaciones** de un conjunto $A = \{a_0, ..., a_{n-1}\}$ a las diferentes aplicaciones biyectivas que pueden establecerse entre el conjunto $\{0, 1, 2, ..., n-1\}$ y el conjunto A.

Como también debe conocerse, el número de permutaciones posibles con un conjunto de cardinal n es $n!$.

La permutación denominada ***perfect shuffle*** (traducido literalmente "naipes perfectamente barajados") (Stone, 1971) se utiliza con mucha frecuencia en las redes de intercomunicación de elementos de proceso. Esta permutación deriva su nombre del proceso de barajar los naipes. Para conseguirlo sobre un mazo de cartas convencional, se corta la baraja en dos partes iguales y después se va tomando una de cada parte para conseguir el mazo barajado. De esta forma, se consigue que todas las cartas que eran adyacentes antes de la operación, estén separadas, al menos, por otra carta. En la figura 4.5 puede verse la permutación sufrida por 8 elementos

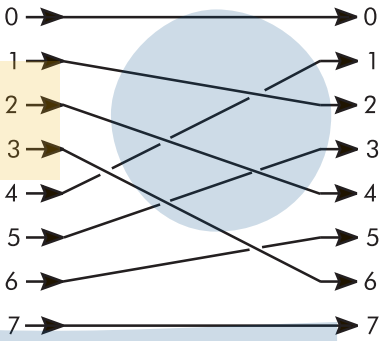


Fig. 4.5. Situación de 8 elementos antes y después de un *perfect shuffle*.

después de efectuar sobre ellos un *perfect shuffle*.

Para formar un *perfect shuffle*, si el número de elementos es potencia de dos, es decir, $n = 2^r$, basta rotar un lugar hacia la izquierda el número de cada elemento en binario (que tendrá r bits), y con ello obtendremos el lugar correspondiente a ese elemento después del *perfect shuffle*.

Para demostrar esta propiedad, dividiremos los elementos del conjunto en dos mitades:

$$A_0 = \{a_0, ..., a_{\frac{n}{2}-1}\} \quad \text{y} \quad A_1 = \{a_{\frac{n}{2}}, ..., a_{n-1}\}$$

En otras palabras, los elementos cuya posición inicial (en binario) comienza por 0, y aquéllos en que comienza por 1.

Para A_0 ($i < n/2$), la forma de construcción de la permutación hace que la nueva situación del elemento i sea:

$$S(i) = 2i \quad \left(i < \frac{n}{2}\right)$$

ya que el *perfect shuffle* se construye intercalando, uno a uno, los elementos de A_1 entre los de A_0.

Por análogas razones, para el segundo conjunto ($i \geq n/2$), tendremos que:

$$S(i) = 2i + C \quad \left(i \geq \frac{n}{2}\right) \qquad [4.8]$$

para algún C que deberemos calcular. Para hacerlo, tomaremos como referencia el primer elemento de A_1 en que

$$S\left(\frac{n}{2}\right) = 1 \Longrightarrow S\left(\frac{n}{2}\right) = 2\left(\frac{n}{2}\right) + C = 1 \Longrightarrow C = 1 - n$$

por tanto, 4.8 se transforma en

$$S(i) = 2i + 1 - n \quad \left(i \geq \frac{n}{2}\right)$$

Resumiendo todo lo dicho anteriormente, tendremos:

$$S(i) = \begin{cases} 2i, & i < \frac{n}{2} \\ 2i + 1 - n, & i \geq \frac{n}{2} \end{cases} \qquad [4.9]$$

Ahora debemos demostrar que estas expresiones son equivalentes a la rotación a la izquierda mencionada anteriormente. En cuanto a la expresión dada para $i < n/2$, es bastante evidente dado que el número de orden de todos los elementos de A_0 comienza por 0; esto significa que la rotación es equivalente a la multiplicación por 2. En cuanto a la expresión para $i \geq n/2$, teniendo en cuenta que el número de orden de estos elementos comienza siempre por un 1, tendremos que la rotación será equivalente a:

Peso del "1" que se pierde por la izquierda "1" que entra por la derecha

$$2i - \overbrace{2^r}^{} + \overbrace{1}^{} = 2i - n + 1$$

porque $2^r = n$, con lo que queda demostrada la equivalencia entre 4.9 y la rotación a la izquierda.

De esta propiedad, se infiere una cualidad muy interesante de la permutación *perfect shuffle* que consiste en que, cuando se efectúa dicha permutación r veces sobre 2^r elementos, regenera el orden original.

Evidentemente, existe una permutación, llamada *perfect shuffle* inverso, que efectúa la operación inversa. Esta permutación también es útil en algunas ocasiones.

La permutación *perfect shuffle* es un caso particular de una clase de permutaciones denominada genéricamente k-***shuffle***. La idea de estas permutaciones se basa también en la acción de barajar los naipes, pero en lugar de dividir la baraja en dos montones, como en el *perfect shuffle*, se divide en k montones, de m cartas cada uno, y, luego, se toma la carta superior de cada montón sucesiva y rotativamente para formar el mazo barajado. Este nuevo orden de las $k * m$ cartas (u objetos, en general) vendrá dada por una permutación a la que denotaremos por S_{k*m}. Formalmente, la permutación k-*shuffle* vendrá dada por:

$$S_{k*m}(i) = \begin{cases} ki \bmod (km-1), & i \in [0, km-1) \\ km-1, & i = km-1 \end{cases} \qquad [4.10]$$

Como fácilmente puede comprobarse, un *perfect shuffle* es lo mismo que un 2-*shuffle*.

Es preciso señalar que la permutación inversa de un k-shuffle de $k * m$ objetos es un m-shuffle de esos objetos. En concreto, el *perfect shuffle* (2-*shuffle*) inverso es un $n/2$-*shuffle*.

Otra permutación bastante utilizada es la **permutación mariposa** (*butterfly*) de k^n elementos de orden i ($0 \leq i < n$) que consiste en intercambiar los dígitos de orden 0 e i del número del elemento en base k. Es decir:

$$M_i^k[(a_{n-1}...a_i...a_0)_{(k}] = (a_{n-1}...a_0...a_i)_{(k}$$

donde M_i^k representa la permutación mariposa de orden i en base k.

Esta permitación tiene la interesante característica de ser simétrica, es decir, que coincide con su inversa. En la figura 4.6 pueden verse las permutaciones mariposa de 2^3 elementos de órdenes 1 y 2.

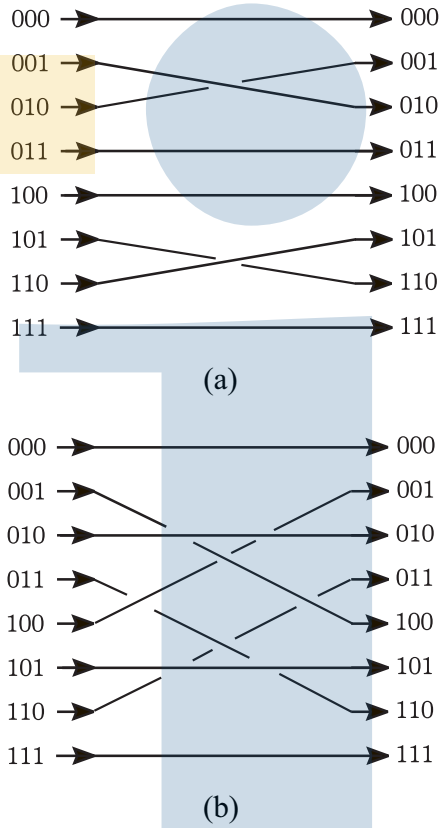


Fig. 4.6. Permutaciones mariposa de 2^3 elementos: (a) de orden 1 y (b) de orden 2.

4.6. Clasificación de las redes de interconexión

El criterio más importante para la clasificación de las redes de interconexión se basa en la rigidez de los enlaces entre los nodos: a este respecto las redes pueden clasificarse en **estáticas** y **dinámicas**. Una red estática se caracteriza porque su topología queda establecida de forma definitiva y estable cuando se instala el sistema; su única posibilidad de modificación es crecer. Por el contrario, una red dinámica puede variar de topología bien durante el curso de la ejecución de los procesos o bien entre la ejecución de los mismos.

Por otra parte, las redes pueden ser jerárquicas o no, lo son si están formadas por una serie de niveles, con diferente número de nodos, dentro de cada uno de los cuales existe *simetría*. La mayoría de las redes jerárquicas suelen ser estáticas, sin embargo, hay algún tipo de topología dinámica que también puede serlo.

4.6.1. Redes de interconexión estáticas

Las redes estáticas emplean enlaces directos fijos entre los nodos. Estos enlaces, una vez fabricado el sistema, son difíciles de cambiar, por lo que la escalabilidad de estas topologías es baja. Las redes estáticas pueden utilizarse con eficiencia en los sistemas en que puede predecirse el tipo de tráfico de comunicaciones entre sus procesadores. Las principales clases de redes de

interconexión estáticas son:

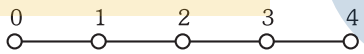
Formación lineal: se trata de una red unidimensional en que los N nodos se conectan cada uno con el siguiente mediante $N-1$ enlaces formando una línea. Esta situación puede verse en la figura 4.7(a). Los nodos interiores tienen grado 2 y los extremos tienen grado 1. El diámetro de esta red es $N-1$ y su ancho de bisección es 1. Esta estructura es la más sencilla, pero no es regular y puede hacerse muy ineficiente para valores altos de N debido a que su diámetro crece linealmente con el número de nodos. Las redes lineales sólo son eficientes cuando tienen muy pocos nodos.

Anillo y anillo cordal: una red en **anillo** se consigue conectando entre sí los nodos extremos de una formación lineal (véase figura 4.7(b)). Esta estructura es simétrica y todos sus nodos tienen grado 2. El diámetro de esta red es $N/2$. Si los enlaces del anillo fueran unidireccionales, su diámetro sería $N-1$. Si se incrementa el grado de cada nodo de la red a 3 o 4, llegaremos a la estructura de **anillo cordal** (figura 4.7(c) y (d)). Según se aumenta el grado de los nodos del anillo cordal, se disminuye su diámetro. El caso más extremo de anillo cordal es la **red totalmente conectada** (figura 4.7(e)), en que el grado de cada nodo es $N-1$, en otras palabras, cada nodo está conectado con todos los demás (es decir su diámetro es 1). El inconveniente de esta topología es la gran complejidad de sus nodos debido a su alto grado (máximo). Un caso especial de anillo cordal, que constituye una buena solución de compromiso entre complejidad y eficiencia, es el denominado ***barrel shifter***, en que el nodo i se conecta con todos los nodos j que cumplen la condición $|i-j| = 2^r, \forall r \in \{0, 1, ..., k-1\}$, donde $k = \log_2 N$. Esta configuración se muestra en la figura 4.7(f). En una red con topología *barrel shifter* el grado de los nodos es $2k-1$ y el diámetro es $\lceil k/2 \rceil$.

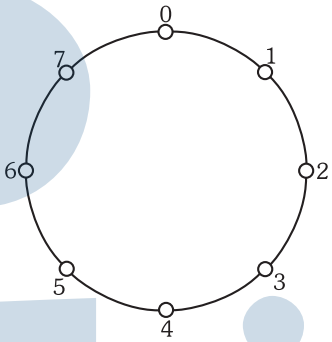
Mallas y toros: una malla k-dimensional tiene l^k nodos donde l es el número de nodos por lado. El grado de los nodos varía entre k para los vértices hasta $2k$ en los nodos más interiores por lo que esta red no es regular y su diámetro es $k(l-1)$. En la figura 4.7(g) se muestra una malla bidimensional con $l = 4$. Esta red de interconexión es muy utilizada en la práctica. Las redes en **toro** son mallas en que sus filas y columnas tienen conexiones en anillo, esto contribuye a disminuir su diámetro. Esta modificación convierte a las mallas en estructuras simétricas y además, reduce su diámetro a la mitad. En la figura 4.7(h) se muestra la malla de la figura 4.7(g) convertida en toro.

Hipercubo: un hipercubo (o más concretamente un n-cubo) es una malla n-dimensional en que se han suprimido los nodos interiores. Según esta definición, un hipercubo de dimensión 1 tendría sólo 2 nodos y, en general, un n-cubo puede formarse interconectando los nodos equivalentes de dos $(n-1)$-cubos (ver figura 4.8). Por ello, el grado de los nodos de un n-cubo es n, lo que hace a esta topología difícilmente escalable. Por todo esto, cuando la dimensión aumenta, se va duplicando el número de nodos, con lo que un n-cubo tendrá 2^n nodos. Los hipercubos han sido bastante utilizados en computadores paralelos, sin embargo, en la actualidad, las máquinas que tenían esta arquitectura han ido evolucionando hacia otras topologías más escalables como se verá a continuación.

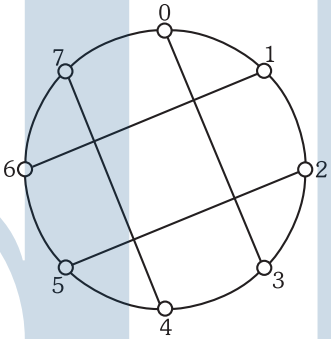
Ciclos hipercubo-conectados (Preparata & Vuellemin, 1979): una red de ciclos n-cubo conectados (abreviadamente n-CCC) es una red n-cubo en que se ha sustituido cada vértice por



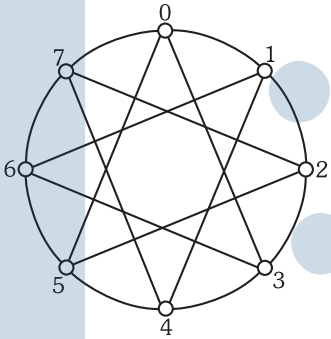
(a) Formación lineal



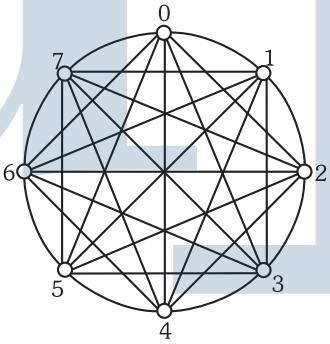
(b) Anillo



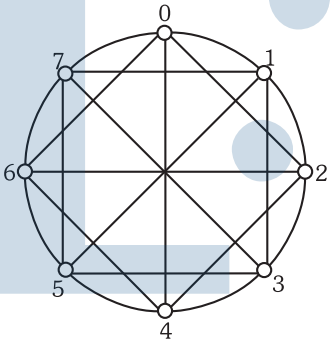
(c) Anillo cordal de grado 3



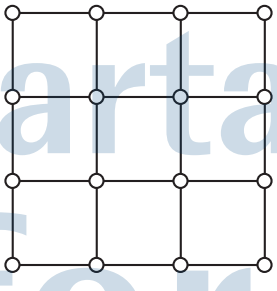
(d) Anillo cordal de grado 4



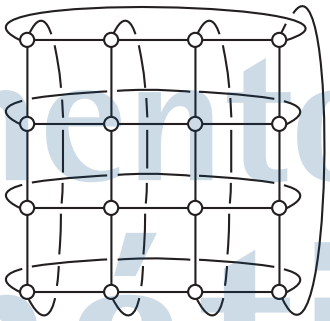
(e) Red completamente conectada



(f) Barrel shifter



(g) Malla



(h) Toro

Fig. 4.7. Topologías de redes estáticas: (a) formación lineal, (b) anillo, (c) y (d) anillos cordales, (e) red totalmente conectada, (f) *barrel shifter*, (g) malla y (h) toro.

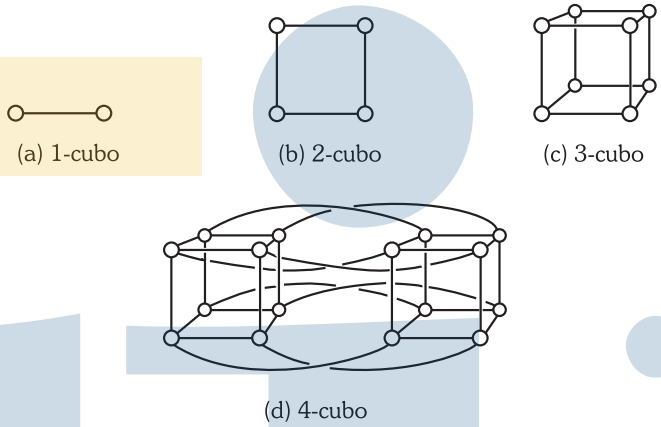


Fig. 4.8. Hipercubos.

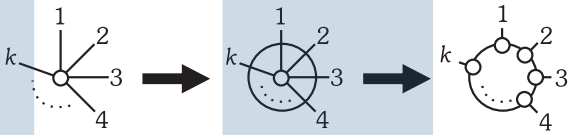


Fig. 4.9. Evolución de un nodo de un n-cubo hacia un ciclo para formar una red de ciclos n-cubo-conectados.

un anillo de n nodos (ver figura 4.9). En este caso, el número de nodos ascenderá a $n2^n$ y el diámetro a $2n$. En la figura 4.10 puede verse una red 3-CCC. Una característica interesante de esta red es que todos sus nodos tienen grado 3 independientemente del valor de n; por ello, este tipo de redes son más escalables que los hipercubos. Para escalar esta topología, aumentando su dimensión, basta poner un nodo más en cada anillo.

n-cubos k-arios: esta topología es una generalización de algunas de las anteriores (Dally, 1990). Un n-cubo k-ario consiste en un cubo n dimensional con k nodos a lo largo de cada dimensión; por otra parte, todas las aristas, tanto exteriores como interiores, tienen estructura de anillo. En la figura 4.11 se muestra una red de esta clase con $n = 3$ y $k = 4$, es decir, un 3-cubo cuaternario. Los anillos, los toros y los hipercubos son casos particulares de n-cubos k-arios (ver ejercicio 4.18). El número total de nodos en una red n-cubo k-aria

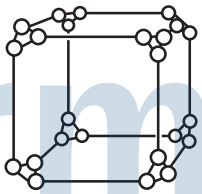


Fig. 4.10. Ciclos 3-cubo-conectados.

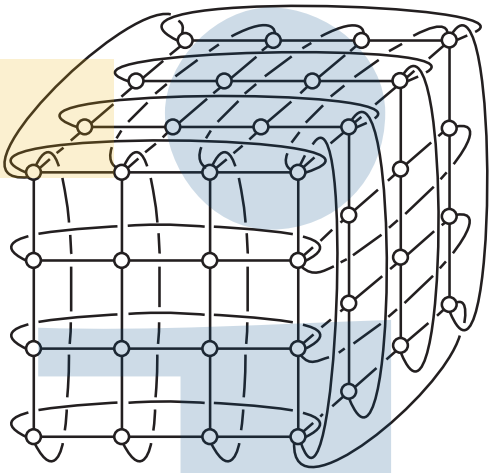


Fig. 4.11. Red 3-cubo cuaternaria.

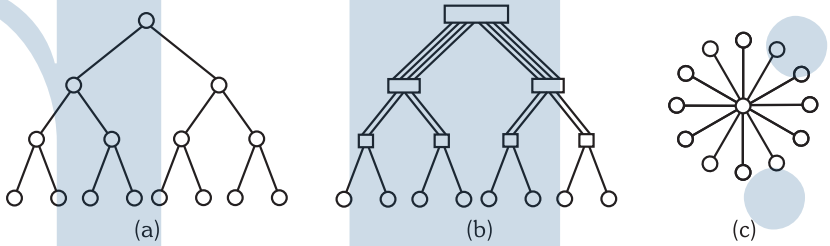


Fig. 4.12. Redes de interconexión jerárquicas: (a) árbol binario, (b) árbol "grueso" (*fat tree*) y (c) estrella.

viene dado por:

$$N = k^n$$

Cada nodo de una red de este tipo puede determinarse completamente por una secuencia de n dígitos en base k, donde cada uno de ellos representa la posición del nodo en cada una de las dimensiones. Estos dígitos serán, por tanto, las **coordenadas** del nodo.

Topologías jerárquicas: el ejemplo más extremo de topología jerárquica es el árbol. Un árbol binario y equilibrado de k niveles tendrá $2^k - 1$ nodos. El grado de los nodos es 3 (excepto para la raíz y los nodos terminales) y el diámetro de esta red es $2(k-1)$ (ver figura 4.12(a)). Las topologías en árbol adolecen de un inconveniente: la raíz puede concentrar el tráfico procedente de muchos nodos, por lo que en ella puede producirse un cuello de botella. Esto puede evitarse procurando dejar en los subárboles información que tenga consistencia propia, si ello fuera posible; de esta forma se disminuirá el tráfico entre dichos subárboles y, por tanto, también por la raíz.

Una modificación de la estructura de árbol es el **árbol grueso** (*fat tree*) (Leiserson, 1985), en que los nodos interiores son conmutadores, y no elementos de proceso, que se comportan como pequeñas centrales telefónicas (ver figura 4.12(b)). La ventaja de esta topología, sobre el árbol convencional, radica en que los conmutadores están diseñados para admi-

Tabla 4.1. Resumen de las características de las redes de interconexión estáticas para N nodos

Tipo	Máximo grado de los nodos	Diametro de la red	Numero de enlaces	Ancho de bisección	Simetría
Lineal	2	$N-1$	$N-1$	1	No
Anillo	2	$\lfloor N/2 \rfloor$	N	2	Sí
Barrel shifter	$2\log_2 N - 1$	$\left\lceil \frac{\log_2 N}{2} \right\rceil$	$\frac{(2\log_2 N-1)N}{2}$	$N + N/2 - 2$	Sí
Totalmente conectada	$N-1$	1	$\frac{N^2-N}{2}$	$N^2/4$	Sí
Malla de 2 dimensiones	4	$2(\sqrt{N}-1)$	$2N - 2\sqrt{N}$	$\sqrt{N}$	No
Toro de 2 dimensiones	4	$2\left\lfloor \frac{\sqrt{N}}{2} \right\rfloor$	$2N$	$2\sqrt{N}$	Sí
Hipercubo	$\log_2 N$	$\log_2 N$	$\frac{N\log_2 N}{2}$	$N/2$	Sí
n-CCC	3	$2n + \lfloor n/2 \rfloor - 1$	$3N/2$	2^{n-1}	Sí
n-cubo k-ario	$2n$	$n\lfloor k/2 \rfloor$	nN	$2k$	Sí
Árbol binario	3	$2(\log_2(N+1)-1)$	$N-1$	1	No
Estrella	$N-1$	2	$N-1$	$\lfloor N/2 \rfloor$	No

tir varios mensajes simultáneamente, por lo que el ancho de bisección es mayor. Esto elimina el inconveniente de los árboles mencionado con anterioridad, ya que el número de enlaces va incrementándose según nos acercamos a la raíz. Este tipo de red, aunque mantiene forma de árbol, en realidad es una red de interconexión dinámica porque los conmutadores cambian las conexiones entre los nodos.

La estructura en **estrella** es un caso específico de árbol con sólo 2 niveles (véase figura 4.12(c)) en que el nodo raíz tiene grado $N-1$ y el resto, grado uno. El diámetro de esta red es 2. Esta topología corresponde a sistemas donde existe un nodo central que hace funciones de supervisor o maestro (por ejemplo, en algunos sistemas paralelos sólo uno de los procesadores dispone de dispositivos de entrada/salida).

En la tabla 4.1 puede verse un resumen de las características de las diferentes redes estáticas.

4.6.2. Redes de interconexión dinámicas

Las redes de interconexión dinámicas son convenientes en los casos en que se desee una red de propósito general ya que son fácilmente reconfigurables. También por eso, este tipo de redes facilitan mucho la escalabilidad. En general, las redes dinámicas necesitan de elementos de conexión específicos como pueden ser árbitros de bus, conmutadores, etc. Las principales topologías de redes dinámicas son las siguientes:

Buses: podríamos definir un **bus** como *un conjunto de líneas que permite comunicar selectivamente un cierto número de componentes o dispositivos de acuerdo a ciertas normas de conexión*. En nuestro caso, los componentes que se conectan son procesadores, bancos de memoria, etc. El inconveniente de un bus es que sólo permite una transferencia al mismo tiempo, por ello, en caso de que haya varias peticiones de comunicación simultáneas, debe haber un **árbitro de bus** que vaya ordenando y dando paso, una a una, a las diferentes peticiones. Este árbitro será un circuito digital que lleve control de los momentos en que el bus está ocupado. Para recalcar más esta forma de funcionamiento, a veces a los buses también se les denomina **buses de tiempo compartido**. Los buses son una forma barata de comunicación que tienen la ventaja de ser muy fácilmente reconfigurables. Sus inconvenientes son su bajo ancho de banda y su gran latencia debida a las esperas que tienen que efectuar las peticiones de comunicación. En la figura 4.13(a) se muestra una estructura de bus. Existen variantes de esta estructura que, incrementando muy poco el coste, pueden mejorar notablemente las prestaciones: estas variantes se basan en disponer de una jerarquía de buses, de forma que en cada uno de ellos pueda establecerse una comunicación en su nivel de forma independiente y simultánea a los demás. Un ejemplo de este tipo de estructura se ilustra en la figura 4.13(b).

Redes de líneas cruzadas o **matriz de conmutación (*crossbar*)**: en esta red, cada nodo está conectado con todos los demás a través de un **conmutador de líneas cruzadas** (***crossbar switch***) en la forma indicada en la figura 4.14. La red de líneas cruzadas puede interpretarse como una central telefónica que conecta los nodos en función de las necesidades de cada momento. Cada conmutador puede proporcionar una conexión dedicada entre cada par de elementos que se quiera conectar. La posición de cada conmutador se cambia dinámicamente según las necesidades del programa. Este tipo de redes se ha empleado para conectar procesadores con módulos de memoria. Un ejemplo de ello es el multiprocesador C.mpp de la Universidad Carnegie-Mellon (Wulf & Bell, 1972). Como fácilmente puede comprenderse, en ese caso, cada módulo de memoria sólo puede responder a una sola petición en cada momento, aunque varios procesadores (si fuera necesario, todos) pueden acceder a diferentes módulos de memoria simultáneamente. También pueden utilizarse las redes de líneas cruzadas para conectar N procesadores con ellos mismos; de esta forma, si se efectúan todas las conexiones posibles, de todas las formas distintas, se podrían construir las $N!$ permutaciones de los N procesadores.

Las principales ventajas de las redes de líneas cruzadas es que no son bloqueantes y son fácilmente escalables. Su mayor inconveniente es que precisa gran número de conmutadores, ya que son necesarios N^2 conmutadores para una red cuadrada de lado N.

Redes multietapa o **MIN** (***multistage interconnection network***): una red de este tipo está formada por una serie de capas de módulos conmutadores $p \times q$. Estos conmutadores pueden cambiar dinámicamente de posición para establecer las conexiones deseadas en cada momento. Las diferentes clases de redes multietapa que se verán a continuación, difieren en el tipo de módulo conmutador empleado y en la forma de interconectarlos (interconexión entre etapas, ISC: *interstage connection*). Un esquema genérico de una red multietapa se muestra en la figura 4.15. La ventaja de las redes multietapa sobre las de líneas cruzadas es el menor número de conmutadores, que es del orden de $N \log_2 N$, aunque varía en función del tipo concreto de red, sin embargo, algunas de ellas son bloqueantes.

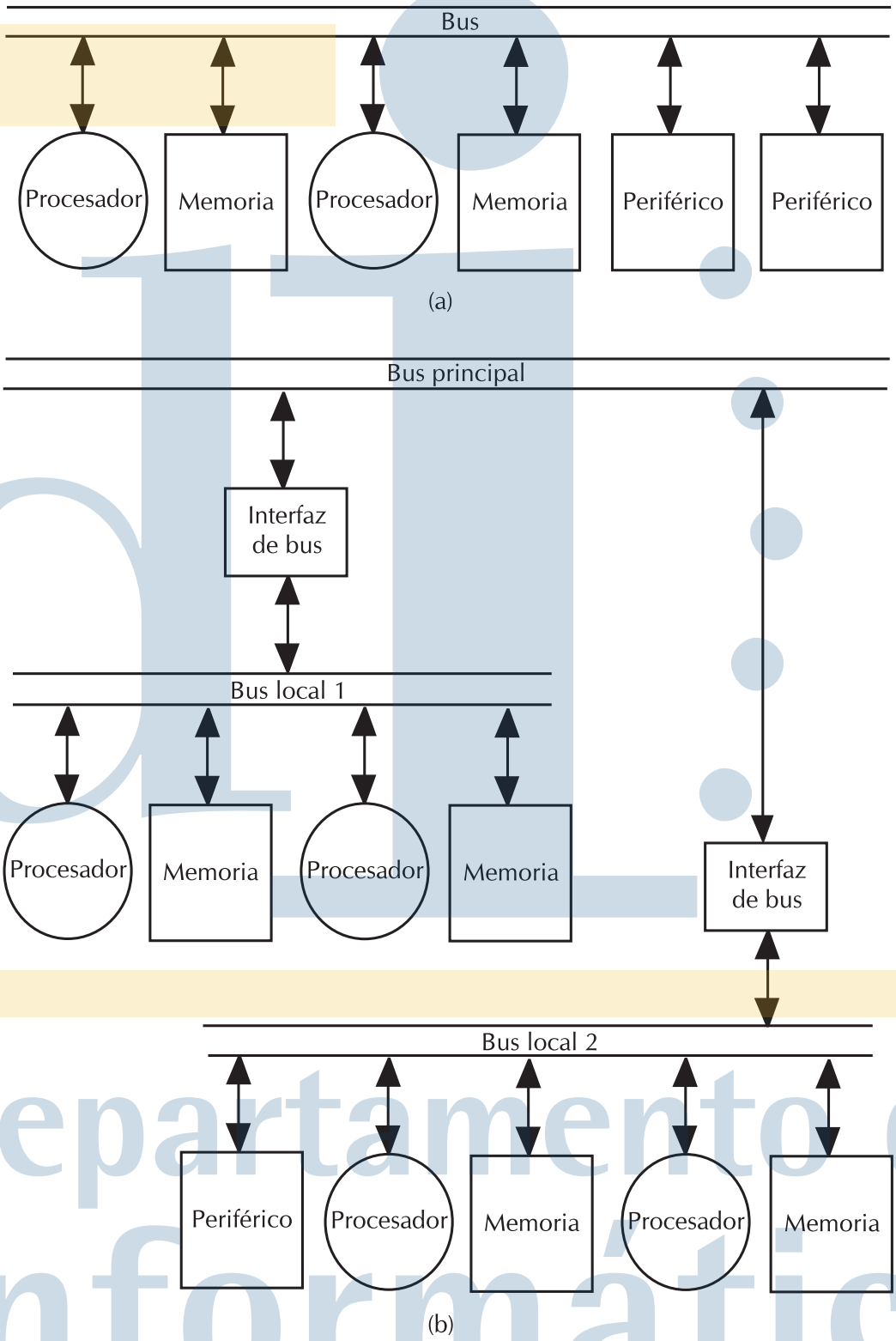


Fig. 4.13. Estructura de bus único (a) y bus jerarquizado (b).

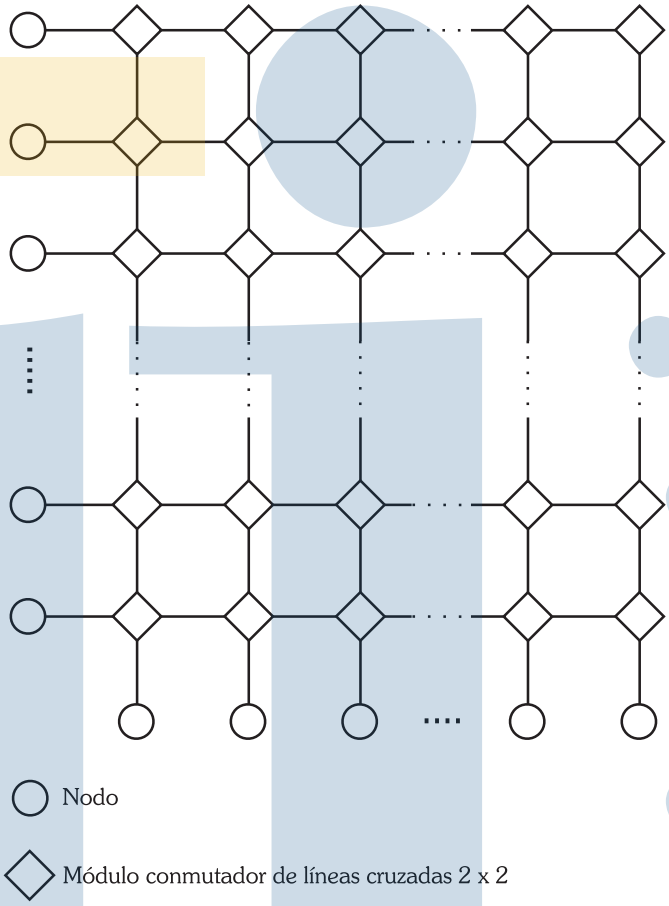


Fig. 4.14. Red de líneas cruzadas (*crossbar*).

Las principales clases de redes multietapa son:

- **Redes omega**: esta red de N entradas está formada por $\log_2 N$ etapas de $N/2$ módulos conmutadores 2×2 cada una. En total, la red tendrá $(N/2)\log_2 N$ conmutadores. Por supuesto, cada conmutador se gobierna de forma independiente a los demás. El patrón de interconexión entre las etapas es un *perfect shuffle* (recordar el apartado 4.5). Un esquema de este tipo de redes se muestra en la figura 4.16.
- **Redes de línea base** (Wu & Feng, 1980): las redes de línea base están formadas por módulos conmutadores de líneas cruzadas, de 2×2, y se pueden generan recursivamente de la siguiente forma: se unen las entradas de cada pareja de conmutadores del último nivel, mediante un *perfect shuffle*, a las salidas de los conmutadores de la etapa anterior. Se agrupan las entradas de doble número de conmutadores en esa etapa, formando otro *perfect shuffle* hacia el nivel anterior. Se repite este proceso hasta llegar a un único *perfect shuffle* que una las salidas de todos los conmutadores del que será el primer nivel. En la figura 4.17(a) se muestra una red de línea base de 16×16 y en (b) puede verse la construcción recursiva de una red de línea base genérica de $N \times N$ a partir de otras dos de $N/2 \times N/2$. Evidentemente, si se mira esta red desde las entradas, se verán permutaciones de *perfect shuffle* inverso.

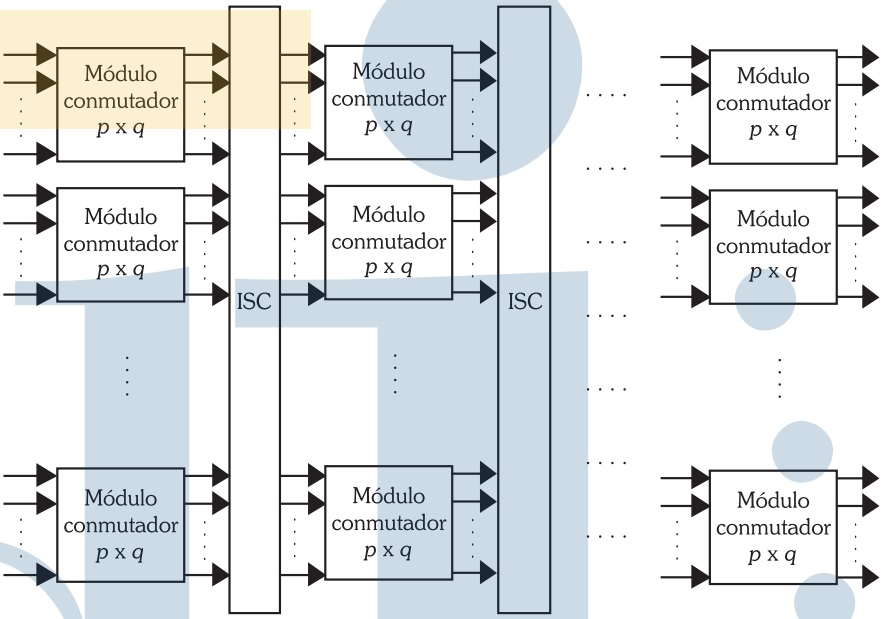


Fig. 4.15. Red genérica multietapa.

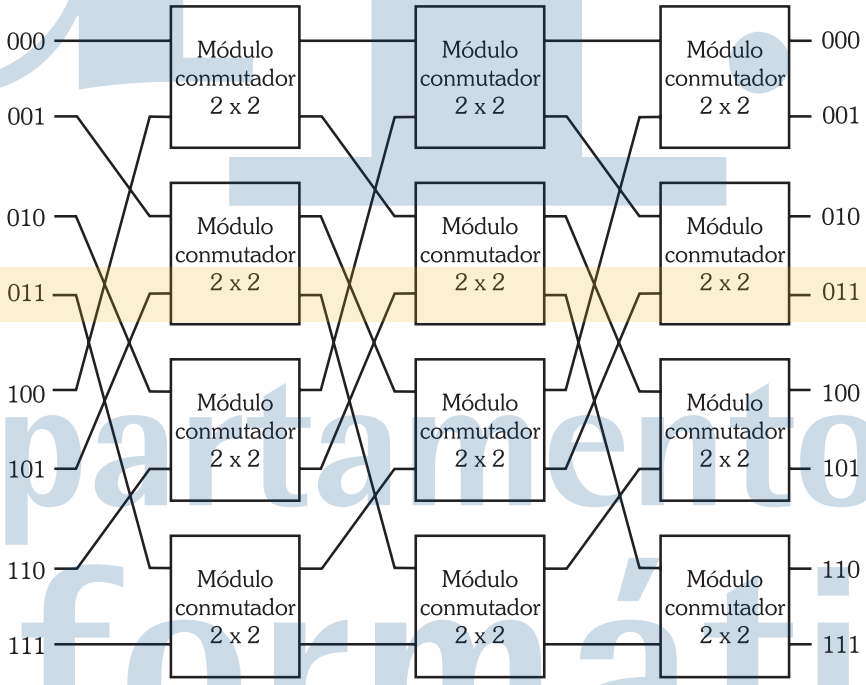


Fig. 4.16. Estructura de una red omega.

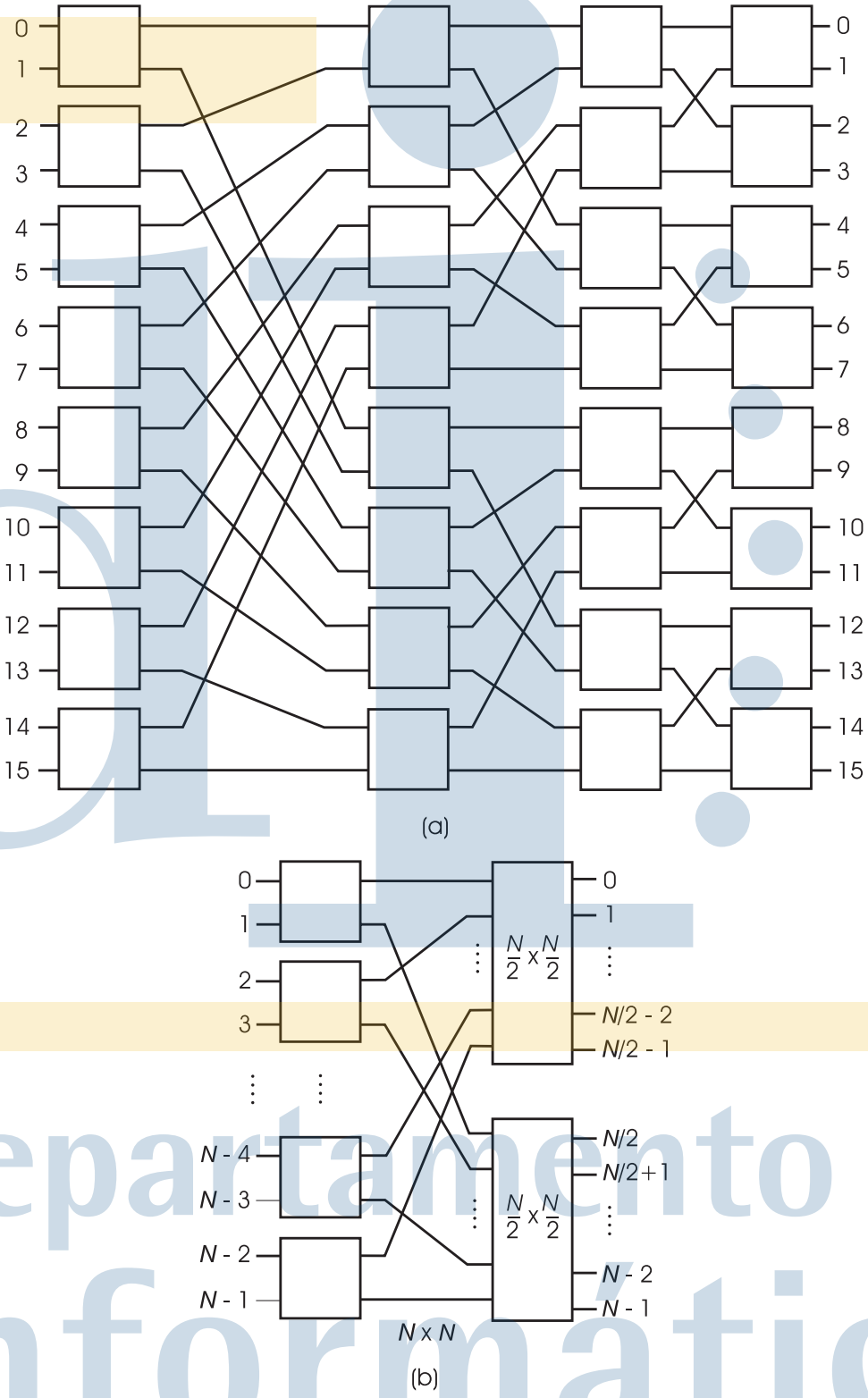


Fig. 4.17. (a) Red de línea base y (b) su construcción recursiva.

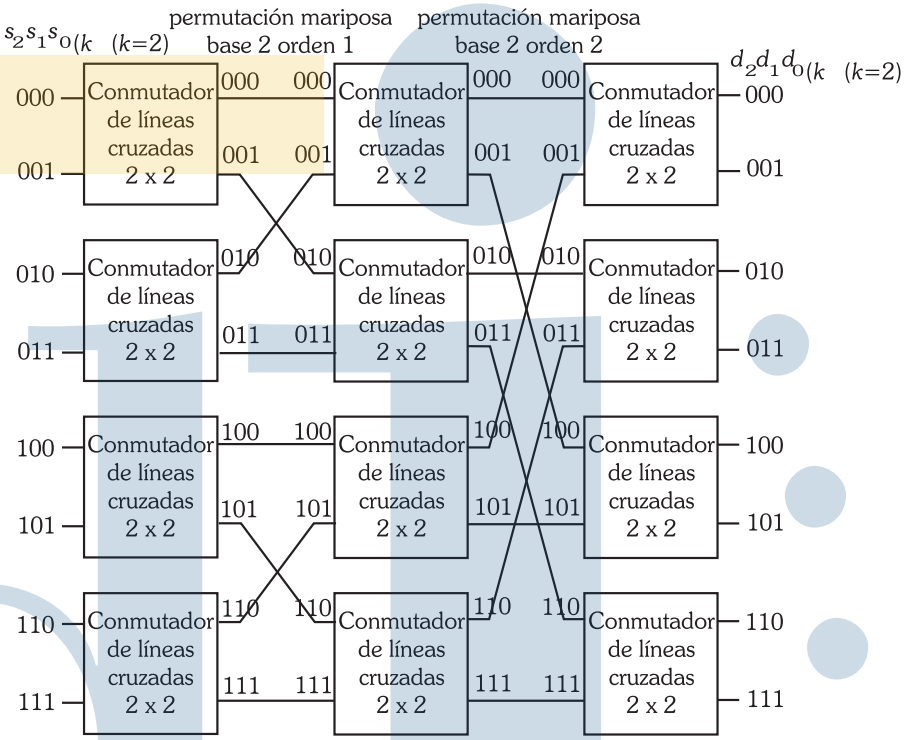


Fig. 4.18. Red mariposa de $2^3 \times 2^3$.

- **Redes mariposa (*butterfly networks*)**: Una red mariposa de $k^n \times k^n$ está formada por n columnas de k^{n-1} conmutadores $k \times k$ unidas por $n-1$ permutaciones mariposa de base k y orden 1 hasta $n-1$. En la figura 4.18 puede verse una red mariposa de $2^3 \times 2^3$.

  Es curioso observar que una red mariposa de $k^n \times k^n$ se puede construir a partir de dos redes mariposa $k^{n-1} \times k^{n-1}$, añadiendo una columna de k^{n-1} conmutadores y uniendo las dos redes con la nueva columna mediante una permutación mariposa de base k y orden $n-1$.

- **Redes Delta**: una red delta se define como una red de interconexión de $a^n \times b^n$ con n etapas formada por módulos conmutadores de líneas cruzadas de $a \times b$ y cuya red de interconexión de etapas es un a-*shuffle*. En la figura 4.19 se muestra una red delta genérica de $a^n \times b^n$. En la figura 4.20 puede verse el ejemplo de una red delta de $2^3 \times 2^3$. Una característica de esta red es que la primera etapa tiene a^{n-1} módulos conmutadores, por tanto, la segunda etapa tendrá $a^{n-2}b$ conmutadores ya que, a cada uno de los conmutadores de la etapa anterior, habrá que multiplicarlo por el número de salidas, b, y dividirlo por el número de entradas, a, para obtener el número de conmutadores de la etapa siguiente. En general, la etapa i tendrá $a^{n-i}b^{i-1}$ y, por tanto, la última etapa (n) tendrá b^{n-1} conmutadores. Esto hace que la red tenga b^n salidas. Se puede demostrar (ver ejercicio 4.25) que el número total de conmutadores de una red delta es:

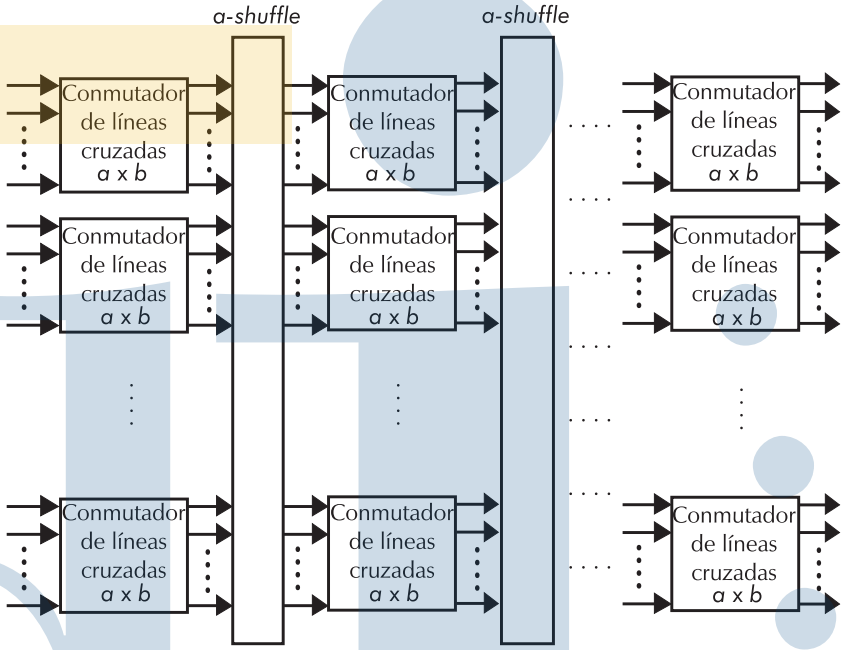


Fig. 4.19. Red delta genérica de $a^n \times b^n$.

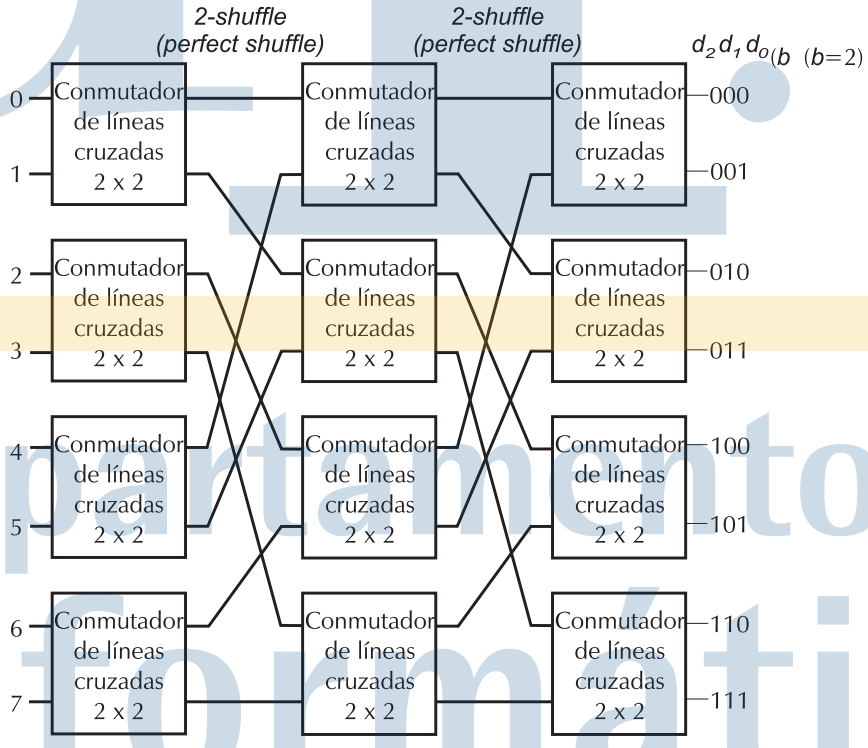


Fig. 4.20. Red delta de $2^3 \times 2^3$.

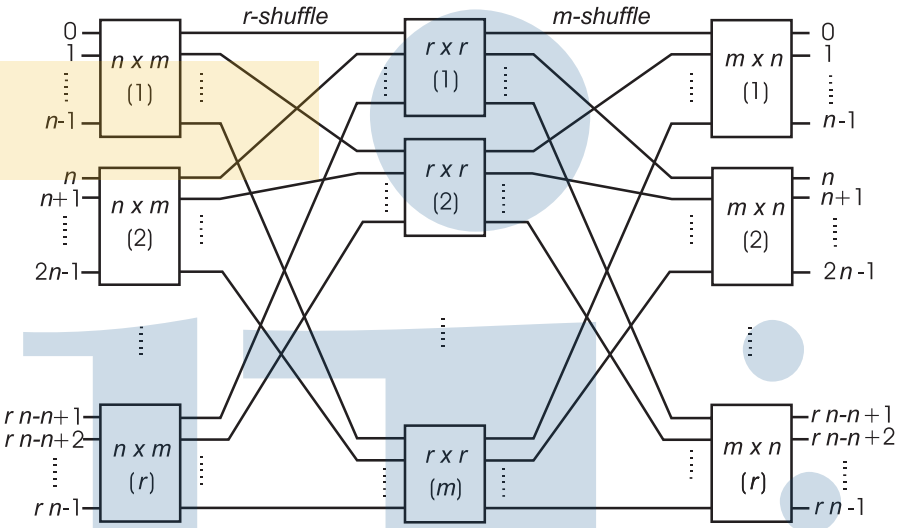


Fig. 4.21. Red de Closs.

$$\begin{array}{ll} \dfrac{a^n - b^n}{a - b} & \text{si } a \neq b \\ na^{n-1} & \text{si } a = b \end{array}$$

- **Redes de Closs**: estas redes constan de tres etapas construidas con conmutadores de líneas cruzadas pero de diferentes tamaños en cada etapa. Cada red de Closs se caracteriza por tres parámetros: n, m y r; el número de entradas de la red es $N = nr$. La primera etapa dispone de r conmutadores de n entradas y m salidas, la segunda etapa contiene m conmutadores de r entradas y r salidas. La conexión entre ambas etapas se efectúa de la forma siguiente: la salida i del conmutador j de la primera etapa se conecta con la entrada j del conmutador i de la segunda (esto en realidad es un r-*shuffle*). La tercera etapa tiene r conmutadores de m entradas y n salidas, siendo las conexiones entre las dos últimas etapas simétricas respecto a las de las dos primeras (es decir forman un m-*shuffle*).

  Una de las ventajas de las redes de Closs es que, dimensionándolas convenientemente, se puede conseguir que sea reacondicionable o, incluso, no bloqueante.

- **Redes de Benes**: Una red de Benes es un red de Closs con $n = 2$ y $m = 2$, en que los conmutadores de la etapa central son, a su vez, redes de Closs con $n = 2$, $m = 2$, siendo r la mitad de la de la red completa. Esto significa que las permutaciones entre etapas son *perfect shuffle* inverso (de la primera a la segunda) y *perfect shuffle*, de la segunda a la tercera.

4.6.3. Resumen comparativo

Como resumen, en la tabla 4.2 se comparan algunas características, de diferentes tipos de redes de interconexión, evaluadas cualitativamente.

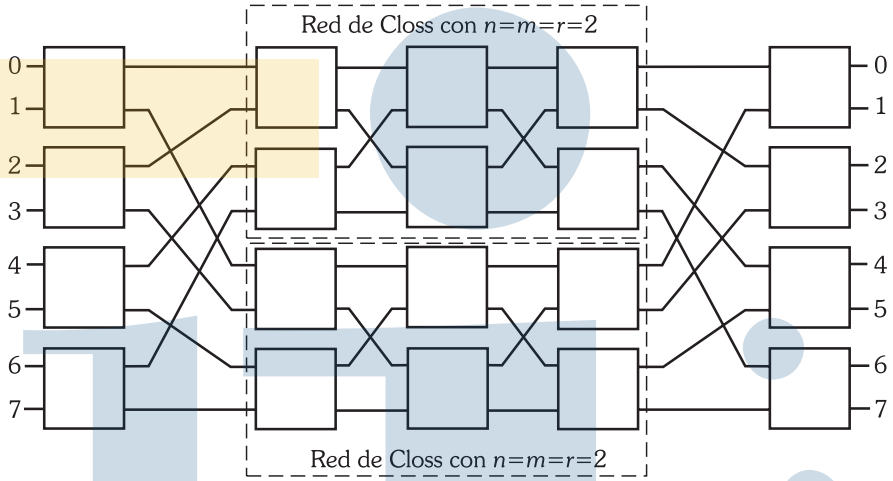


Fig. 4.22. Red de Benes.

4.7. Métodos de encaminamiento

Los métodos de encaminamiento son mecanismos, hardware o software, que permiten establecer la ruta entre los nodos origen y destino de una comunicación. El algoritmo de encaminamiento también debe efectuar la elección de la ruta cuando haya varias posibles, también debe gestionar los conflictos entre las informaciones que quieran tomar el mismo camino. La facilidad en el encaminamiento puede ser una de las razones para elegir uno u otro tipo de red. Estudiaremos los métodos de encaminamiento en algunas de las redes estudiadas con anterioridad.

4.7.1. Encaminamiento en redes hipercubo

Para entender el encaminamiento en las redes de tipo k-cubo es necesario saber como se numeran los nodos en este tipo de red. Lo haremos de forma recursiva partiendo de un 1-cubo. Un 1-cubo tendrá dos nodos a los que numeraremos con 0 y 1. Cada vez que se añada una nueva dimensión, se duplicará el número de nodos existentes: numeraremos los nuevos nodos con el mismo número que los anteriores añadiendo por la izquierda un nuevo bit con valor 1. Se procederá sucesivamente de esta forma para numerar los nodos de un hipercubo de cualquier dimensión.

Tabla 4.2. Resumen comparativo de diferentes redes de interconexión

	Hipercubo	Bus	Líneas cruzadas	Multietapa
Costo	Medio	Bajo	Alto	Medio
Velocidad	Media	Baja	Alta	Alta
Complejidad	Media	Baja	Alta	Media
Escalabilidad	Media	Alta	Media	Media

Sabiendo esto, el algoritmo de encaminamiento es sencillo: Se comparan los números de los nodos origen y destino de la comunicación. A partir de ahí, se envía el mensaje por los enlaces de las direcciones que corresponden a los bits diferentes en ambos números.

4.7.2. Encaminamiento en redes n-CCC

El encaminamiento en estas redes es muy parecido al de las redes hipercubo. La diferencia estriba en que en cada vértice hay que recorrer un enlace por el anillo, para cambiar de dimensión, y en el anillo final hay que llegar hasta el nodo deseado por el lado más corto.

4.7.3. Encaminamiento en redes omega

El principio del encaminamiento en las redes omega radica en que, como se estudió con anterioridad, la permutación *perfect shuffle* conecta cada nodo con el que resulta de rotar su número un lugar a la izquierda y, por otra parte, si nos fijamos en el primer nivel, cada conmutador intercambia los nodos que difieren en el bit de mayor orden: esto equivale a complementar ese bit de cara al encaminamiento. Reuniendo ambos hechos se llega fácilmente al algoritmo de encaminamiento. Para efectuarlo, se parte del nodo origen, esto nos llevará a un módulo conmutador. Este módulo se invertirá si los números de los nodos origen y destino, difieren en el bit de mayor orden. Esto nos conducirá a otro conmutador que se invertirá si los números de ambos nodos difieren en el bit siguiente, se continuará así hasta llegar a la última etapa en que alcanzaremos el nodo destino.

4.7.4. Encaminamiento en redes delta

En una red delta el algoritmo de encaminamiento tiene una característica curiosa: sólo depende del número del nodo de destino: En una red delta de $a^n \times b^n$, si se pone el número del nodo de destino en base b, partiremos del nodo origen y haremos que el primer conmutador conecte la entrada que proviene del nodo origen con la salida correspondiente al valor del dígito más significativo del número del nodo destino (escrito en base b), procederemos así con sucesivos dígitos en las siguientes etapas por las que el paquete de información vaya pasando (véase la figura 4.20).

4.7.5. Encaminamiento en redes de linea base

Para efectuar el encaminamiento en redes de línea base se debe proceder según los pasos siguientes:

1. Se invierte el orden de los bits del número del nodo origen (en binario).
2. Se hace la operación OR EXCLUSIVO del resultado del paso anterior con el número del nodo destino, también en binario.

3. El resultado de la operación anterior nos indicará si cada conmutador se dejará en conexión directa (cuando el bit sea 0) o en conexión cruzada (cuando el bit sea 1), respectivamente.

Bibliografía y referencias

DALLY, W. 1990. Performance Analysis of k-ary n-cube Interconnection Networks. *IEEE Transactions on Computers*, **39**(6).

HWANG, K. 1993. *Advanced Computer Architecture: Parallelism, Scalability, Programmability*. McGraw-Hill.

INDURKYA, B, STONE, H.S., & XI-CHENG, L. 1986. Optimal partitioning of randomly generated distributed programs. *IEEE Transactions on Software Engineering*, **12**(3).

LEISERSON, C.E. 1985. Fat-Trees: Universal Networks for Hardware-Efficient Super-Computing. *IEEE Transactions on Computers*, **34**(10).

ORTEGA, J. ET AL. 2005. *Arquitectura de Computadores*. Thomson.

PREPARATA, F.P., & VUELLEMIN, J.E. 1979. The Cube-Conected Cycles: A Versatile Network for Parallel Computation. *In: Proceedings of 20 Symposium Foundations Computer Science*.

STONE, H.S. 1971. Parallel Processing with a Perfect Shuffle. *IEEE Transactions on Computers*, **20**(Feb.).

STONE, H.S. 1990. *High-performance Computer Architecture*. 2 edn. Addison-Wesley.

TABAK, D. 1990. *Multiprocessors*. Prentice-Hall International.

WU, C.L., & FENG, T.Y. 1980. On a class of Multistage Interconnection Networks. *IEEE Transactions on Computers*, **29**(8).

WULF, W.A., & BELL, C.G. 1972. C.mpp-A Multi-miniprocessor. *In: Proceedings of Fall Joint Computer Conference*.

CUESTIONES Y PROBLEMAS

4.1 Supongamos que, en un sistema multiprocesador con 16 procesadores, se ejecutan 50 tareas que se comunican completamente entre ellas. Calcular la relación R/C a partir de la cual es mejor utilizar todos los procesadores.

4.2 Supongamos un sistema multiprocesador en que se ejecutan 50 tareas. Sabiendo que, para esas tareas, $R = 50$, $C = 4$ y la carga de la comunicación depende linealmente del número de procesadores, calcular el número óptimo de procesadores para este caso.

4.3 Desarrollar el modelo óptimo para los sistemas de comunicación entre procesadores (apartado 4.2.4) suponiendo una dependencia lineal del tiempo de comunicaciones con el número total de procesadores.

4.4 Sea un sistema con dos procesadores en que se ejecutan 10 tareas que pueden repartirse entre ambos procesadores. Supongamos que la carga de cálculo de cada tarea es de 2 y que la carga de comunicaciones es de 1, si las tareas se ejecutan en diferentes procesadores, y es nula si se ejecutan en el mismo. Suponiendo que cada tarea necesita comunicarse con todas las demás y que los tiempos de cálculo y comunicación se pueden superponer completamente: calcular la distribución de tareas entre ambos procesadores para que el tiempo total de ejecución sea mínimo.

4.5 a) Resolver el problema anterior pero suponiendo que los tiempos de cálculo y comunicación no se pueden superponer en absoluto.

b) Si pudiera cambiarse la influencia de las comunicaciones (C): ¿Qué valor tendría que adquirir para que el punto de mejor rendimiento cambiara?

4.6 Construir una permutación *perfect shuffle* con 32 objetos.

4.7 a) Construir una permutación 4-*shuffle* con 32 objetos.

b) ¿Cuál es el valor de k para el que un k-*shuffle* de esos 32 objetos sea la permutación inversa de la anterior?

4.8 Construir una permutación mariposa de 3^2 elementos de orden 1.

4.9 Diseñar la estructura interna de un módulo conmutador 2×2, unidireccional y para un bit.

4.10 Repetir el problema anterior para el caso bidireccional.

4.11 Diseñar la estructura interna de un conmutador de líneas cruzadas 2×2, unidireccional y de un bit.

4.12 Repetir el problema anterior para el caso bidireccional.

4.13 ¿Cuántos estados posibles puede tener un conmutador de líneas cruzadas 4×4?

4.14 ¿Cuántos estados posibles puede haber en un conmutador unidireccional 4×4?

4.15 Dibujar una red *barrel shifter* con 16 nodos.

4.16 Dibujar la estructura de una red 4-CCC (ciclos 4-cubo conectados)

4.17 Comprobar pormenorizadamente los datos de la tabla 4.1.

4.18 Tomando las redes en anillo, toro e hipercubo como n-cubos k-arios, calcular los valores de n y k para cada uno de estos tipos de red suponiendo que cada una de ellas tiene 16 nodos.

4.19 Sea un sistema paralelo con múltiples procesadores donde cada procesador tiene necesidad de comunicarse con cada uno de los demás en la misma medida. Si en este sistema se empleara una red de interconexión en árbol binario:

a) ¿Cuáles serían los puntos calientes?

b) ¿Sería adecuada la topología propuesta para este caso?¿Por qué?

4.20 a) Dibujar el esquema de una red omega para unir 16 módulos de memoria con otros tantos procesadores.

b) Señalar los conmutadores recorridos para comunicar el módulo de memoria 2 con el procesador 6.

c) Suponiendo que se mantiene la comunicación anterior, encontrar un camino para comunicar el módulo de memoria 4 con el procesador 7.

d) ¿Qué demuestra el resultado anterior?

e) ¿Qué habría que hacer para comunicar el módulo de memoria 0 con todos los procesadores?

4.21 a) Dibujar la estructura de una red 5-cubo.

b) Encontrar un camino que una los nodos 5 y 31.

4.22 Dibujar el esquema de una red delta de $4^2 \times 3^2$.

4.23 a) Dibujar el esquema de una red de línea base para conectar 32 módulos de memoria con otros tantos procesadores.

b) Encontrar el camino para comunicar el módulo de memoria 23 con el procesador 8.

4.24 a) Dibujar el esquema de una red delta para unir 27 procesadores con otros 27 módulos de memoria.

b) Encontrar el camino que une el procesador 5 con el módulo de memoria 8.

4.25 Demostrar que el número total de conmutadores de una red delta es:

$$\frac{a^n - b^n}{a - b} \quad \text{si } a \neq b$$

$$na^{n-1} \quad \text{si } a = b$$

4.26 Dibujar una red mariposa de $3^3 \times 3^3$.

4.27 Dibujar el esquema de una red de Benes para unir 16 procesadores con el mismo número de módulos de memoria.

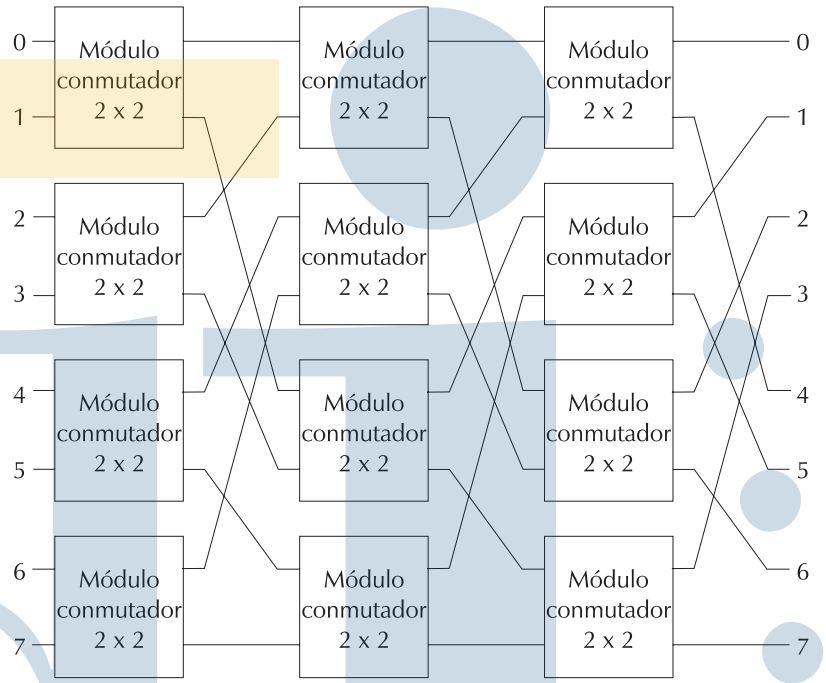


Fig. 4.23. Estructura de una red flip.

4.28 Una red flip es una red multietapa con la estructura que se muestra en la figura 4.23.

a) Encontrar un algoritmo de encaminamiento para este tipo de red.

b) Si se empleara una red de este tipo para comunicar procesadores con módulos de memoria, encontrar el camino para que el procesador 2 escriba un dato en el módulo de memoria 6.

c) Suponiendo que el enlace anterior ya está establecido, encontrar el camino que una el procesador 3 con el módulo de memoria 4.

d) Supuesto que siguen funcionando los enlaces anteriores, encontrar el camino entre el procesador 4 y el módulo de memoria 7.

e) ¿Se puede deducir alguna característica de la red a partir de los resultados anteriores?

4.29 a) Dibujar una red delta de 8×27.

b) Supongamos que esa red comunica 8 procesadores con 27 módulos de memoria. Encontrar razonadamente el camino para conectar el procesador 5 con el módulo de memoria 15.

4.30 Sea una memoria de 16 Mg. distribuida en módulos de 1 Mg. Se quiere conectar 16 procesadores a esa memoria mediante una red omega. Explicar el camino que es necesario recorrer para que el procesador 5 lea un dato en la dirección de memoria BF4C40H.

4.31 Sea una memoria de 64 Mg. distribuida en módulos de 4 Mg. Se quiere conectar 9 procesadores a esa memoria mediante una red delta. Explicar el camino que es necesario recorrer para que el procesador 6 lea un dato en la dirección de memoria 0AB5400H.