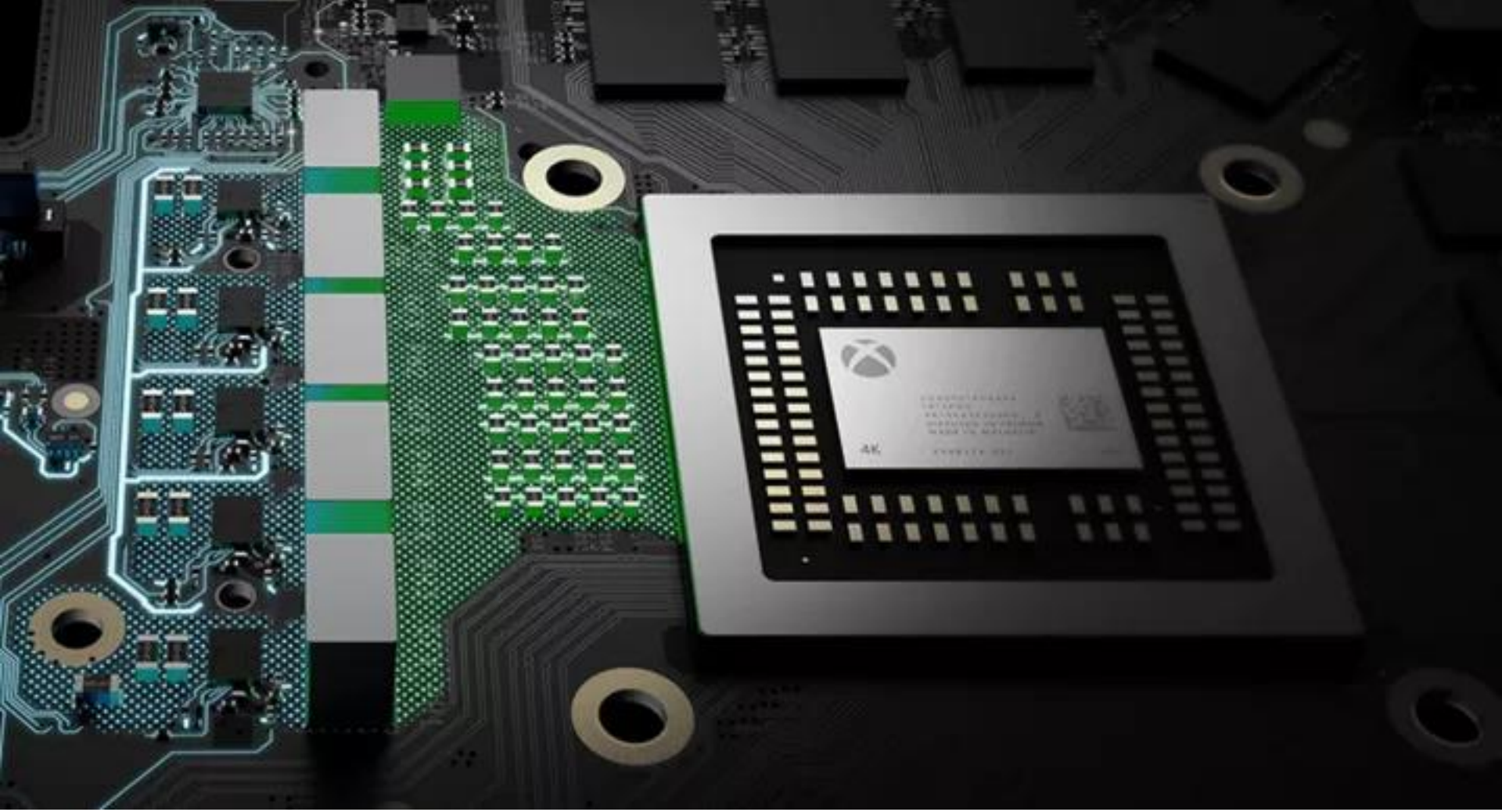


PROGRAMACIÓN en PARALELO

FUNDAMENTOS

COMPUTO en PARALELO

tiene

DISEÑO de PROGRAMAS: Ideas clave

paso

1 Particionamiento

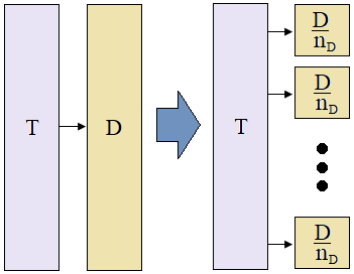
si se necesita o no

2 Coordinación

tipo

Paralelismo de **DATOS**

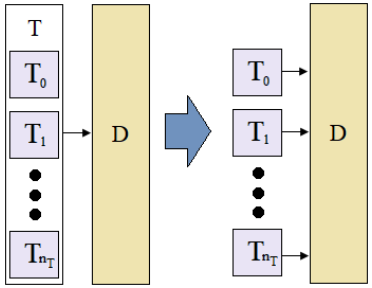
Descomposición del DOMINIO



tipo

Paralelismo de **TAREAS**

Descomposición del FUNCIONAL



en

Comunicación

es

Transferencia de Datos entre Tareas

en

Sincronización

causa

Necesidad de **Cooperación**

hay

Dependencia entre tareas

en

Balance de Carga

es

Cada Procesador debe tener la misma cantidad de trabajo

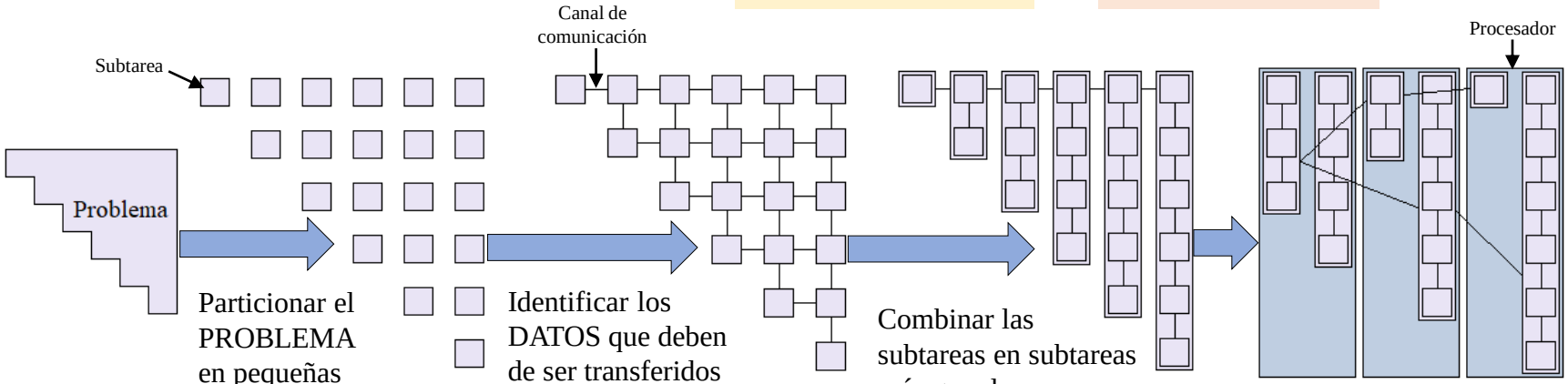
COMPUTO en PARALELO

tiene

DISEÑO de PROGRAMAS: Método de Foster

paso

1 Particionamiento → 2 Comunicación → 3 Unión [Sincronización] → 4 Mapeo [Balance de Carga]



Particionar el PROBLEMA en pequeñas subtareas

Identificar los DATOS que deben de ser transferidos entre cada par de subtareas A y B en términos de un canal de comunicación que conecta A y B

Combinar las subtareas en subtareas más grandes:

- regla: Si la subtarea B depende de la subtarea A entonces unirlas: A→B
- razón: ∞ subtareas → sobrecarga de comunicación
- reduce: Cantidad de DATOS a ser transmitidos entre subtareas
- disminuye

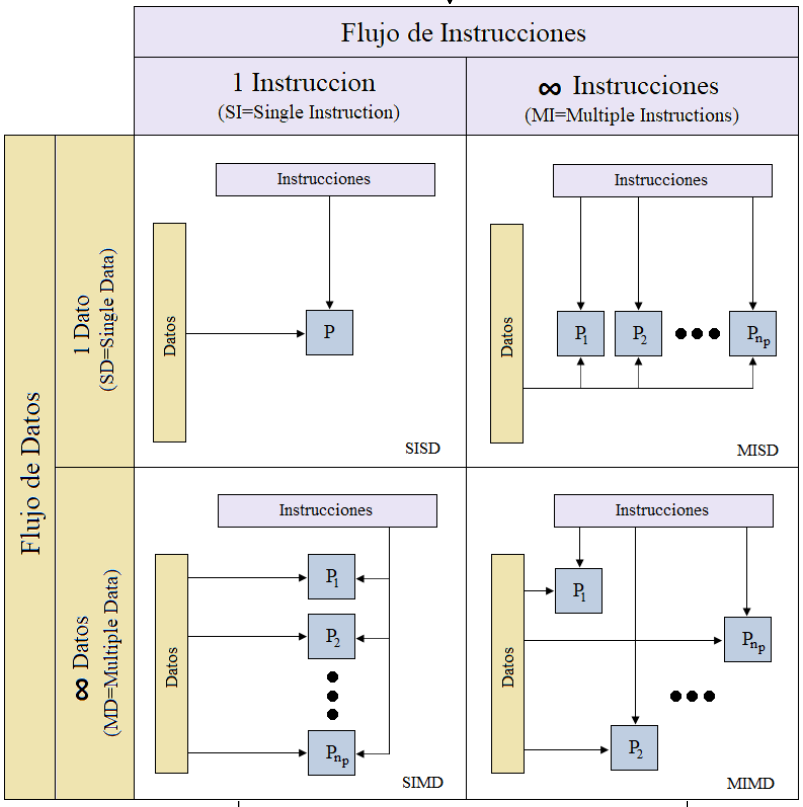
Asignar las subtareas a cada uno de los procesadores de tal forma que se distribuya la misma cantidad de trabajo por procesador y se minimice la comunicación entre procesadores.

COMPUTO en PARALELO

tiene

CLASIFICACIÓN: Procesamiento (Taxonomía de Flynn)

es



característica

uso en

Paralelismo de Datos

uso en

GPU=Graphics PU
TPU=Tensor PU
VPU=Vector PU

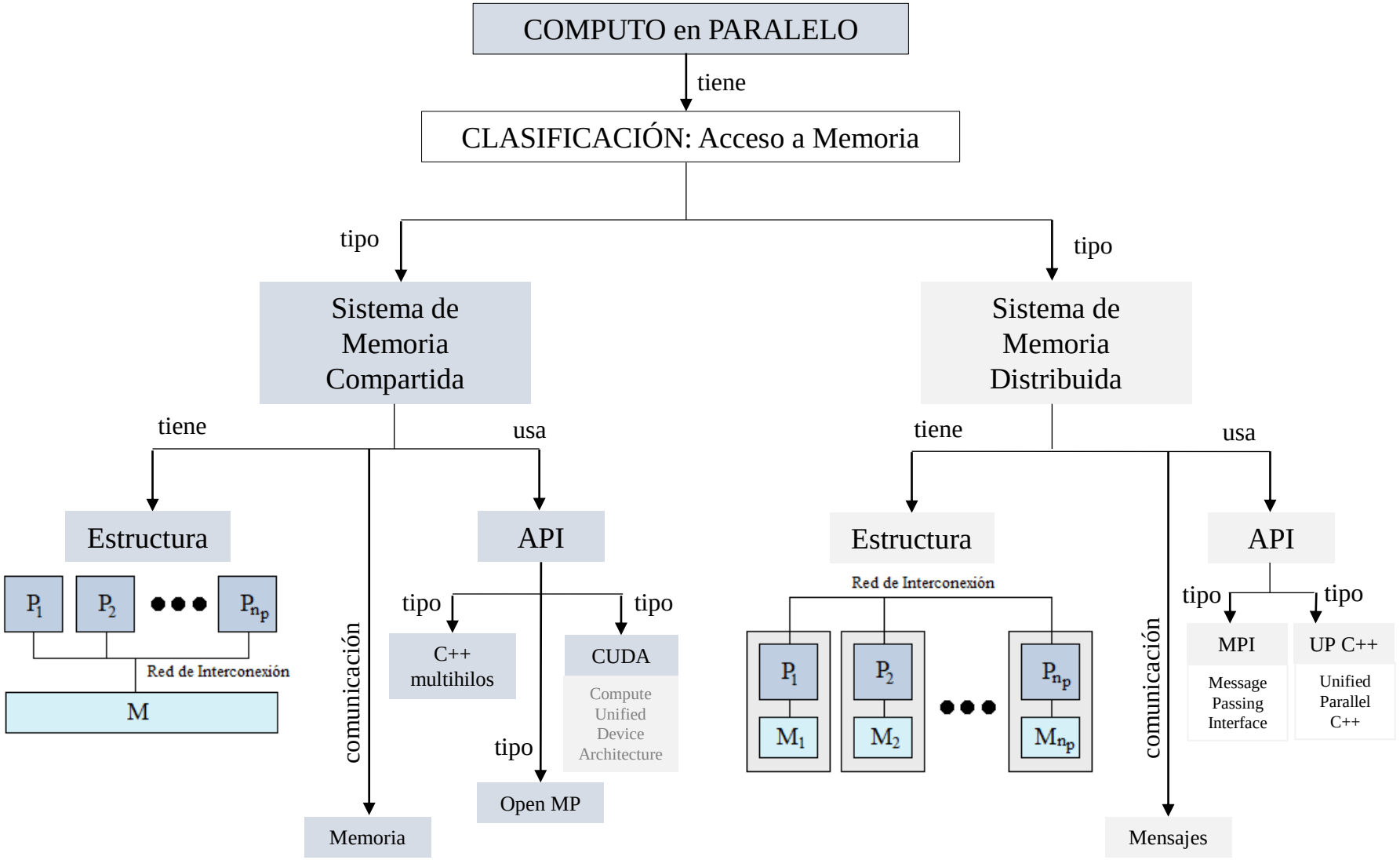
$P_1, P_2, \ldots P_{np}$ son síncronos

característica

uso en

Procesador Multinúcleo

$P_1, P_2, \ldots P_{np}$ son síncronos



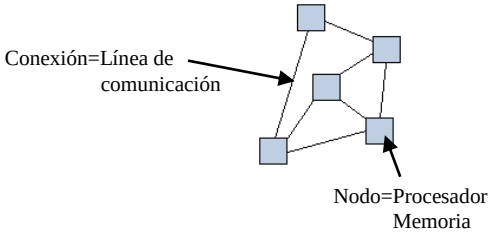
NOTA: P_i=*i*-esima Unidad de Procesamiento
M_i=*i*-esima Memoria
n_p=Numero de Procesadores

RED de INTERCONEXIÓN

es

GRAFO

Conjunto de conexiones entre 2 o más elementos



Determina el nivel de

Paralelización de la comunicación

es

Transferencia simultanea de varios mensajes entre diferentes pares de nodos

definición

Ancho de Banda

Razón a la cual el destino recibe los Datos después de haber recibido el primer byte

Ancho de Banda

Número de bytes en Δt

$$B = \left[\frac{n_b}{\Delta t}\right]$$

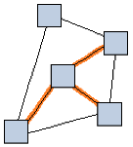
Intervalo de tiempo (s)

definición

N_G=Grado

es

Número máximo de nodos vecinos



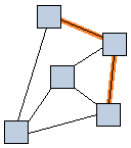
$N_G = 3$

definición

$\downarrow N_D$=Diámetro

es

La máxima distancia de entre todas las distancias mínimas entre pares de nodos



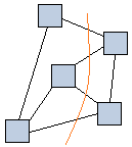
$N_D = 2$

definición

$\uparrow N_{AB}$=Ancho de Bisección

es

Número mínimo de aristas a remover para dividir la red en 2 partes iguales



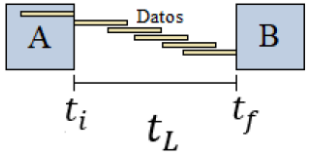
$N_{AB} = 4$

definición

t_L=Tiempo de Latencia

es

Tiempo que transcurre entre la transmisión del primer byte de los Datos en A al tiempo t_i hasta la recepción del primer byte de los Datos en B al tiempo t_f

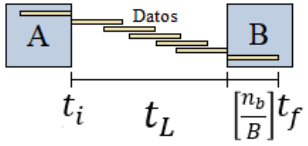


definición

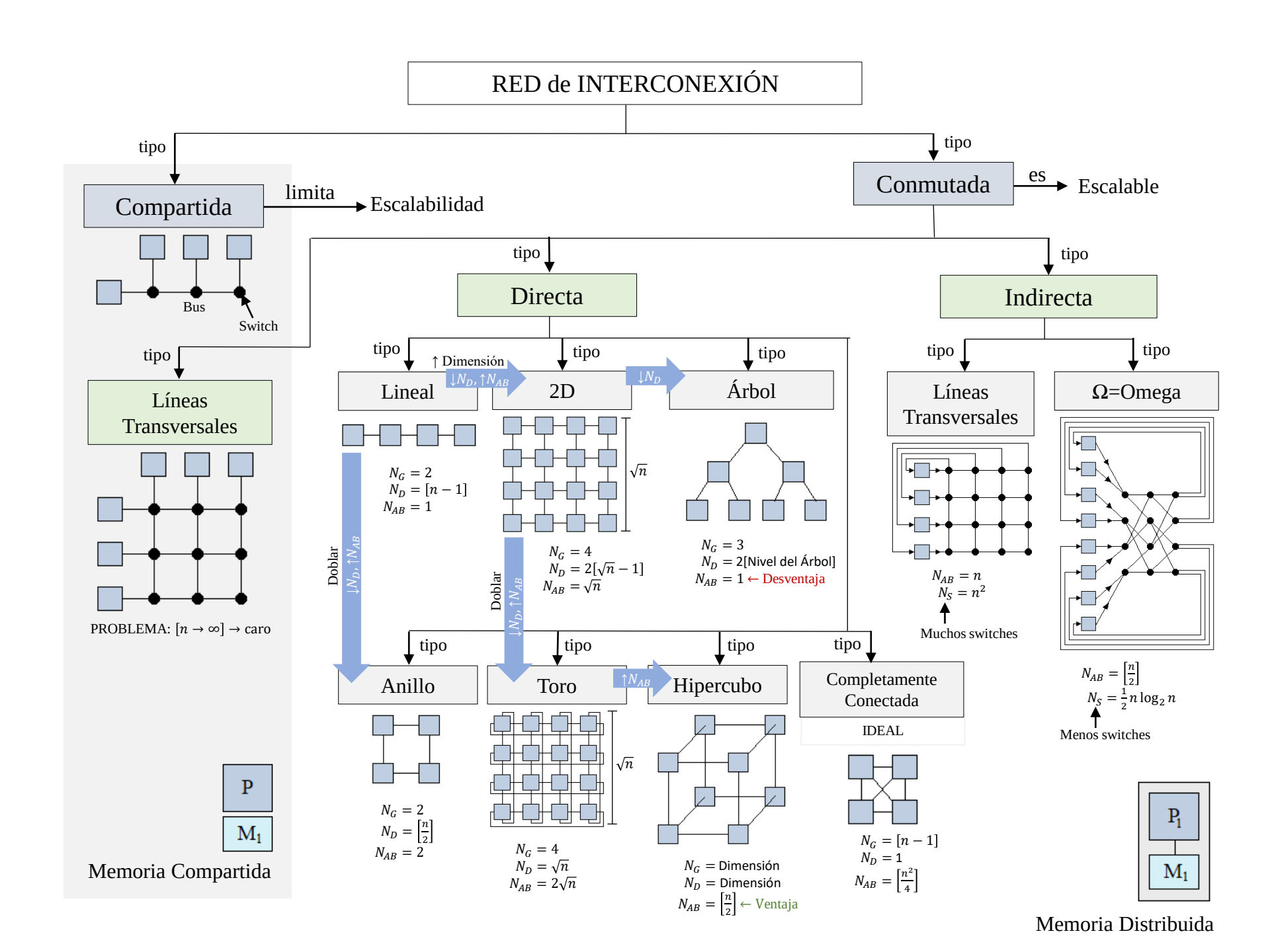
t_T=Tiempo de Transmisión de n_b bytes

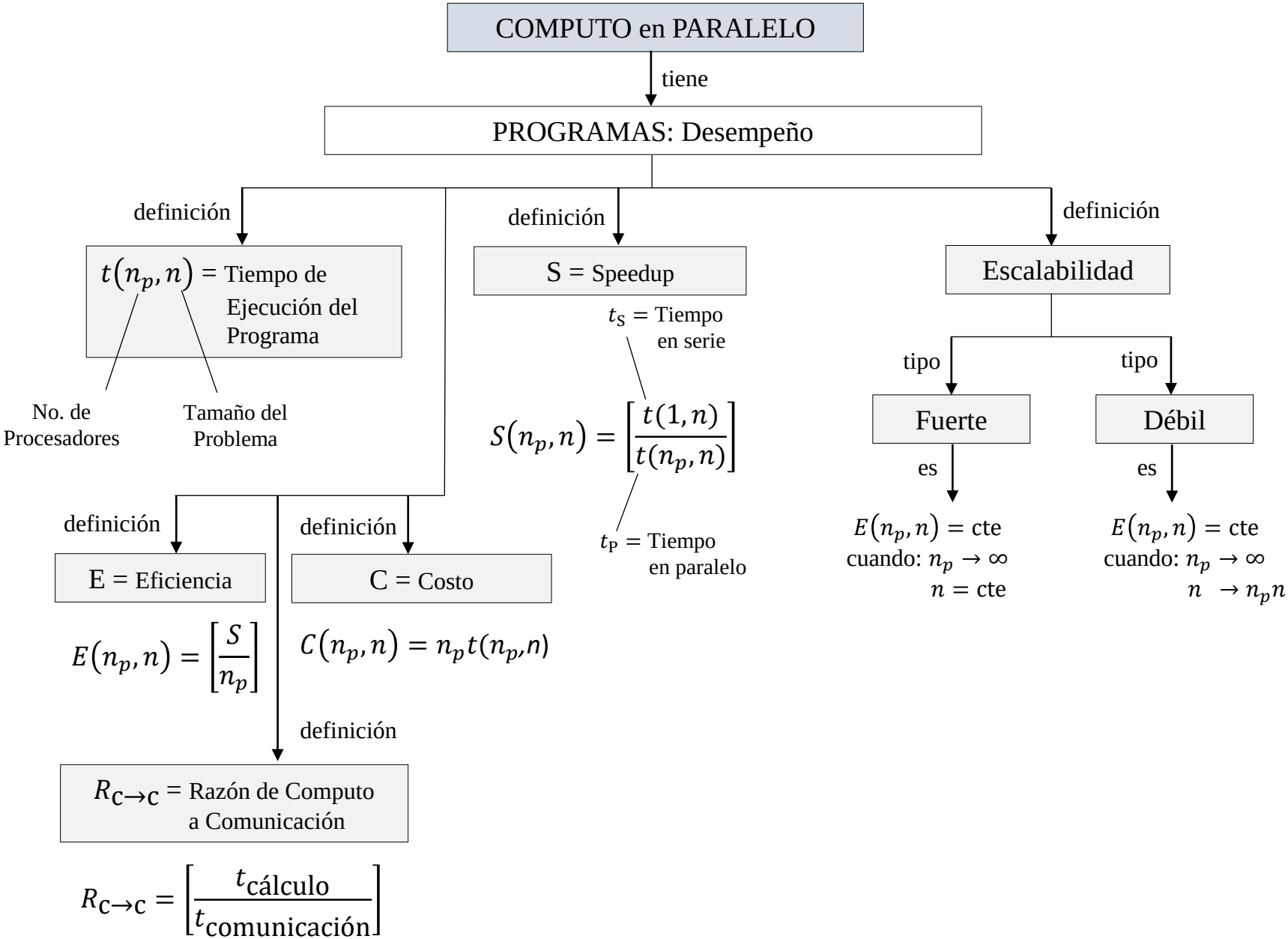
es

Tiempo que transcurre entre la transmisión del primer byte de los Datos en A al tiempo t_i hasta la recepción del ultimo byte de los Datos en B al tiempo t_f



$$t_T = t_L + \left[\frac{n_b}{B}\right]$$


RED de INTERCONEXIÓN
tipo
tipo
Compartida
limita
Escalabilidad
Conmutada
es
Escalable
Bus
Switch
tipo
tipo
Directa
Indirecta
tipo
tipo
tipo
tipo
tipo
Líneas Transversales
↑ Dimensión
↓N_D, ↑N_{AB}
Lineal
2D
↓N_D
Árbol
Líneas Transversales
Ω=Omega
PROBLEMA: $[n \to \infty] \to$ caro
$N_G = 2$
$N_D = [n-1]$
$N_{AB} = 1$
$\sqrt{n}$
$N_G = 4$
$N_D = 2[\sqrt{n}-1]$
$N_{AB} = \sqrt{n}$
$N_G = 3$
$N_D = 2$[Nivel del Árbol]
$N_{AB} = 1$ ← Desventaja
$N_{AB} = n$
$N_S = n^2$
Muchos switches
$N_{AB} = \lceil \frac{n}{2} \rceil$
$N_S = \frac{1}{2} n \log_2 n$
Menos switches
Doblar
↓N_D, ↑N_{AB}
Doblar
↓N_D, ↑N_{AB}
tipo
tipo
tipo
tipo
Anillo
Toro
↑N_{AB}
Hipercubo
Completamente Conectada
IDEAL
$N_G = 2$
$N_D = \lceil \frac{n}{2} \rceil$
$N_{AB} = 2$
$\sqrt{n}$
$N_G = 4$
$N_D = \sqrt{n}$
$N_{AB} = 2\sqrt{n}$
$N_G =$ Dimensión
$N_D =$ Dimensión
$N_{AB} = \lceil \frac{n}{2} \rceil$ ← Ventaja
$N_G = [n-1]$
$N_D = 1$
$N_{AB} = \lceil \frac{n^2}{4} \rceil$
P
M_1
Memoria Compartida
P_1
M_1
Memoria Distribuida


COMPUTO en PARALELO
tiene
PROGRAMAS: Desempeño
definición
definición
definición
$t(n_p, n)$ = Tiempo de Ejecución del Programa
No. de Procesadores
Tamaño del Problema
S = Speedup
t_S = Tiempo en serie
$S(n_p, n) = \left[\frac{t(1,n)}{t(n_p,n)}\right]$
t_P = Tiempo en paralelo
Escalabilidad
tipo
tipo
Fuerte
Débil
es
es
$E(n_p, n)$ = cte
cuando: $n_p \to \infty$
n = cte
$E(n_p, n)$ = cte
cuando: $n_p \to \infty$
$n \to n_p n$
definición
definición
E = Eficiencia
C = Costo
$E(n_p, n) = \left[\frac{S}{n_p}\right]$
$C(n_p, n) = n_p t(n_p, n)$
definición
$R_{c \to c}$ = Razón de Computo a Comunicación
$R_{c \to c} = \left[\frac{t_{cálculo}}{t_{comunicación}}\right]$

COMPUTO en PARALELO

tiene

PROGRAMAS: Desempeño

definición

Tiempo de Ejecución Ideal en Paralelo

$$t_{\mathrm{p}}^{\mathrm{Ideal}} = \left[\frac{t_{\mathrm{S}}}{n_{\mathrm{p}}}\right]$$

definición

Tiempo de Sobrecarga

$$t_{\mathrm{sc}} = \left[t_{\mathrm{p}} - t_{\mathrm{p}}^{\mathrm{Ideal}}\right]$$

en

Memoria Compartida

es

Secciones Críticas

$$\uparrow \left[\begin{matrix}\text{secciones}\\ \text{críticas}\end{matrix}\right] \rightarrow \uparrow [t_{\mathrm{sc}}]$$

en

Memoria Distribuida

es

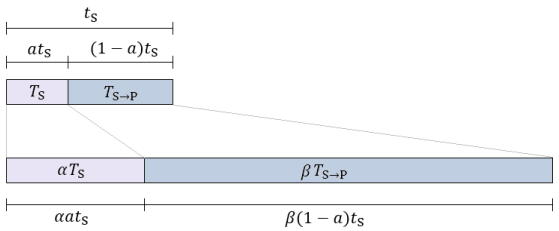
Comunicación

$$\uparrow [\text{Datos}] \rightarrow \uparrow [t_{\mathrm{sc}}]$$

definición

Ley General

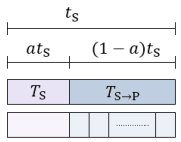
$$S = \left[\frac{a + \frac{\beta}{\alpha}[1-a]}{a + \frac{\beta}{\alpha}\left[\frac{1-a}{n_p}\right]}\right]$$

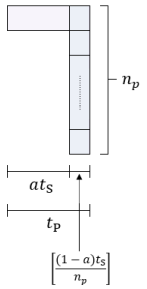


$\left[\frac{\beta}{\alpha}\right]=1$

Ley de Amdahl

$$S = \left[\frac{1}{a + \left[\frac{1-a}{n_p}\right]}\right]$$

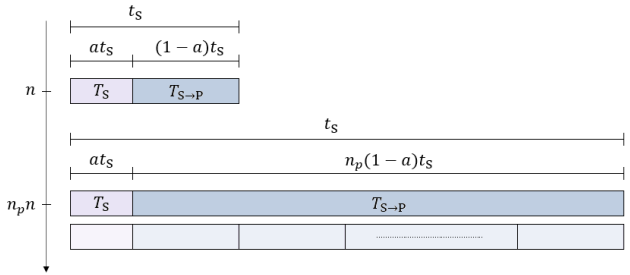


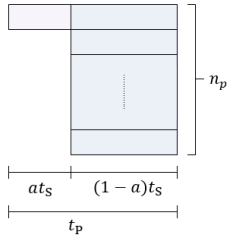


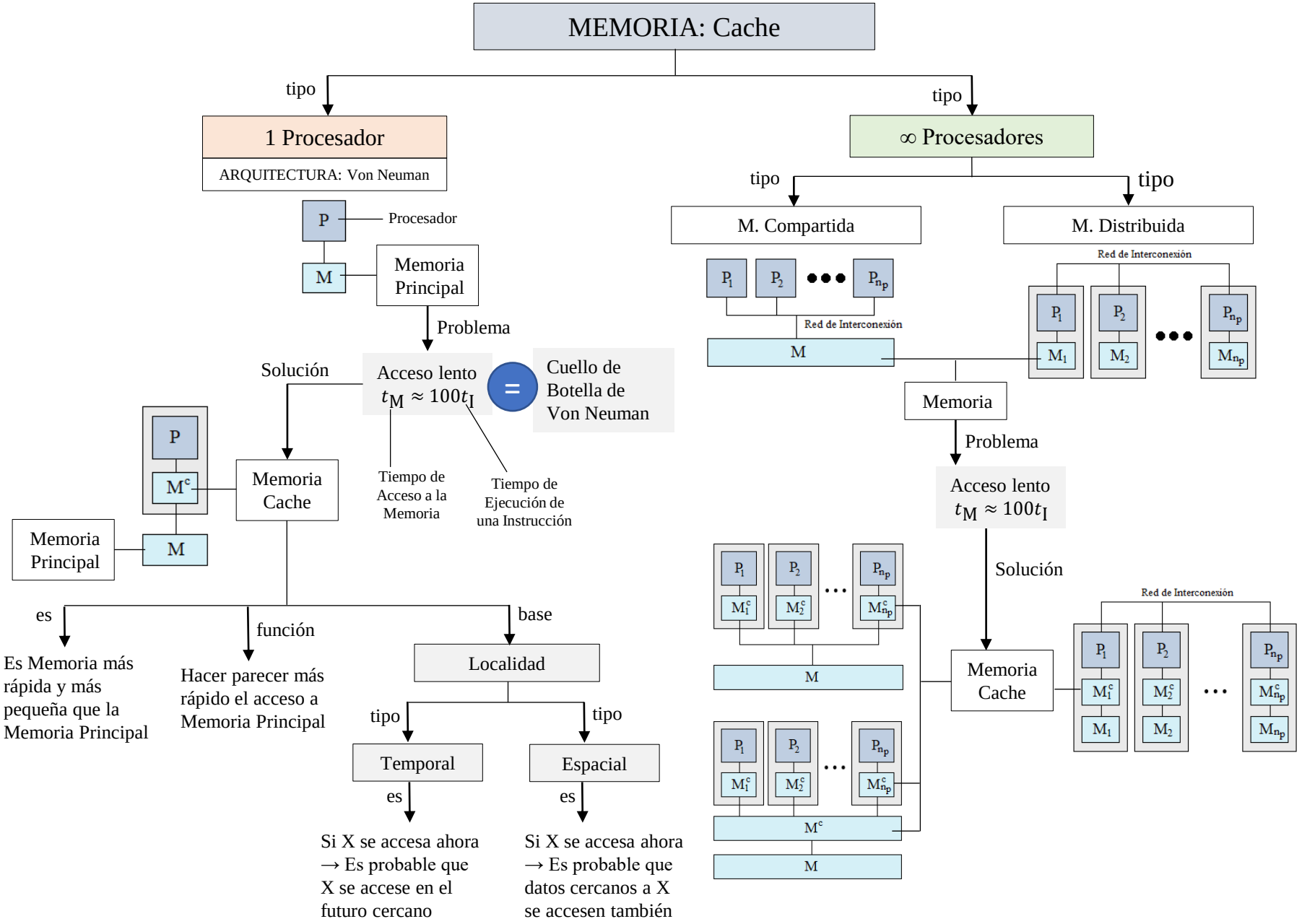
$\left[\frac{\beta}{\alpha}\right]= n_p$

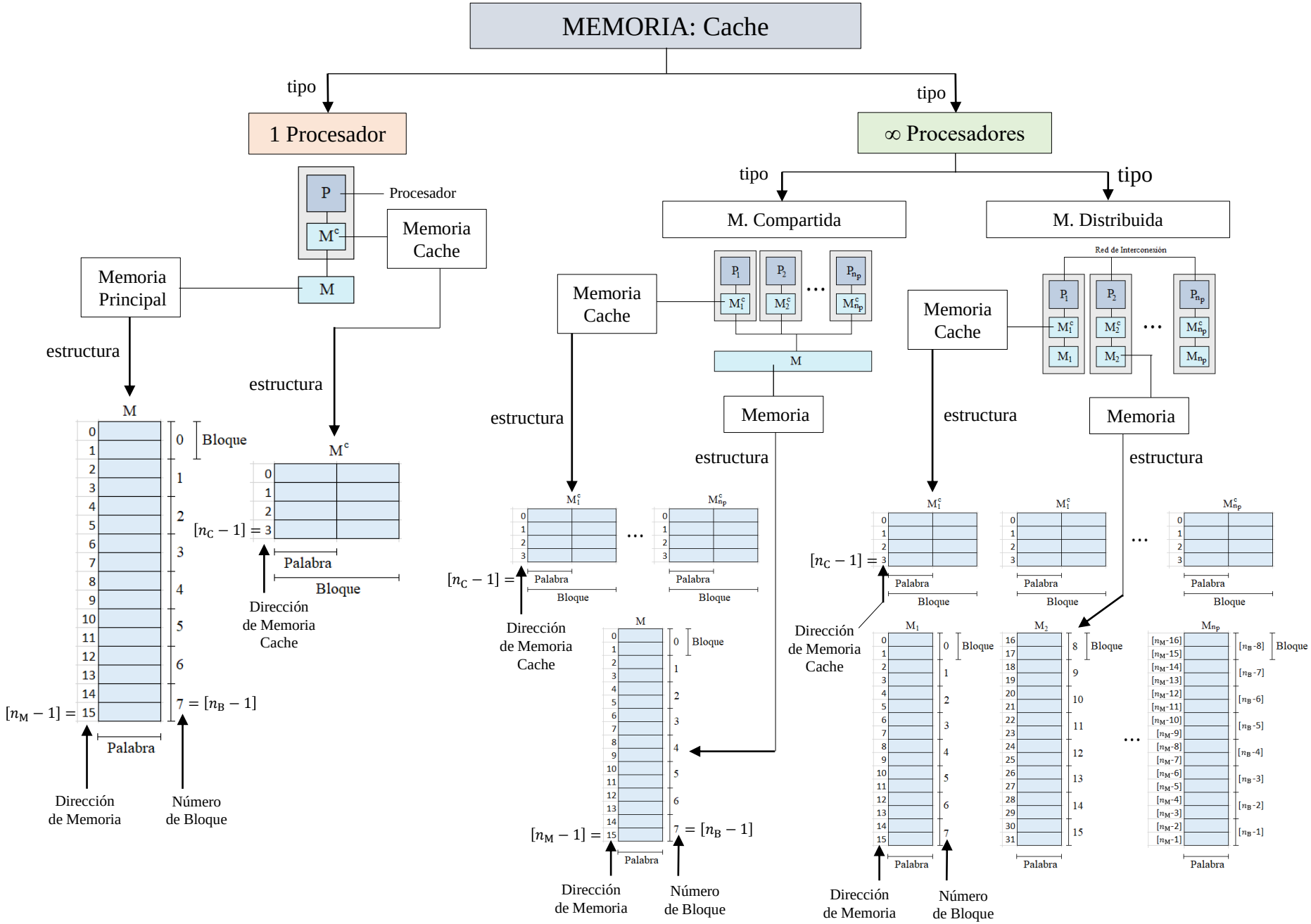
Ley de Gustafson

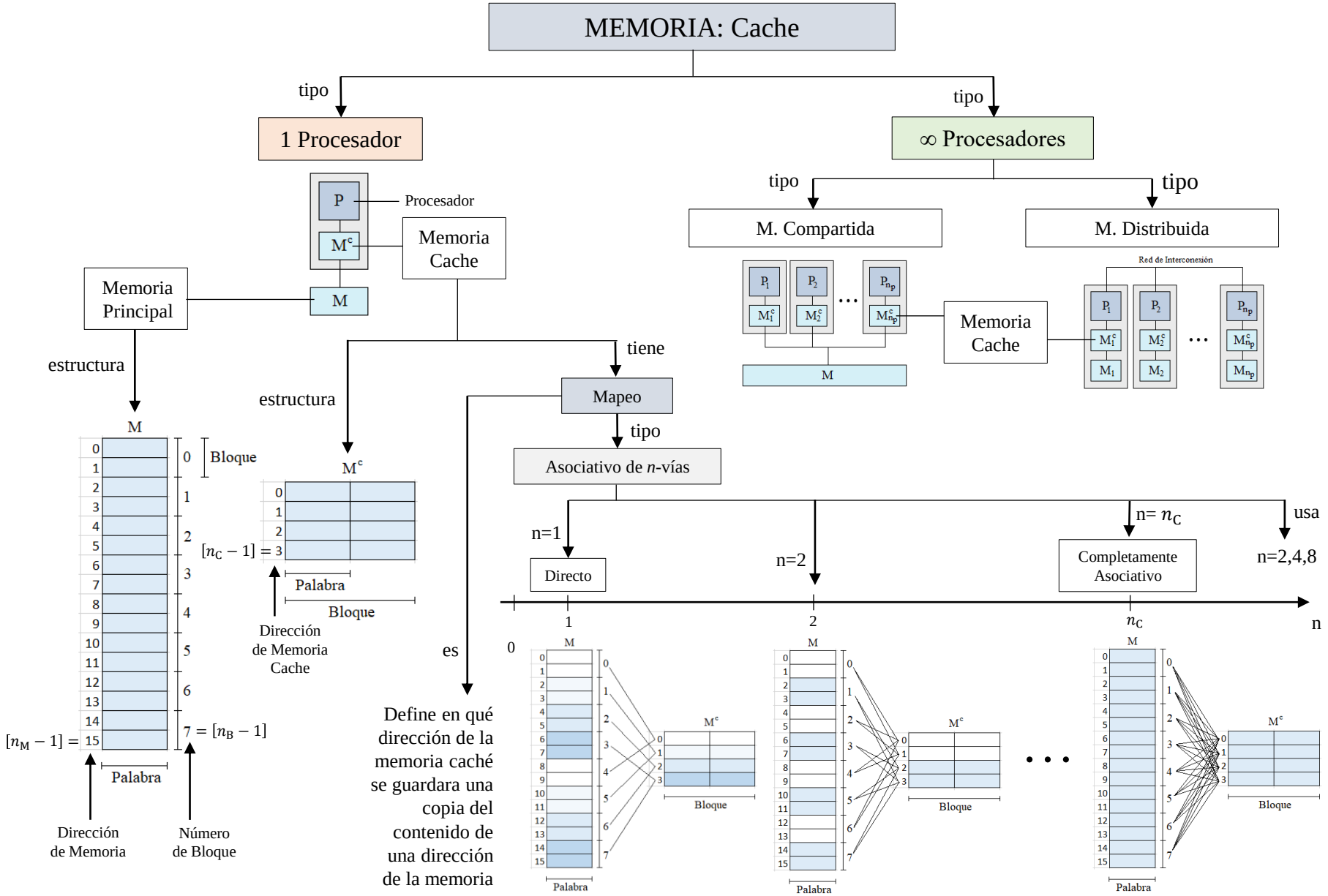
$$S = [1-a]n_p + a$$






MEMORIA: Cache
tipo
1 Procesador
ARQUITECTURA: Von Neuman
P
Procesador
M
Memoria Principal
Problema
Solución
Acceso lento
$t_M \approx 100t_I$
=
Cuello de Botella de Von Neuman
Tiempo de Acceso a la Memoria
Tiempo de Ejecución de una Instrucción
P
M^c
M
Memoria Cache
Memoria Principal
es
Es Memoria más rápida y más pequeña que la Memoria Principal
función
Hacer parecer más rápido el acceso a Memoria Principal
base
Localidad
tipo
Temporal
es
Si X se accesa ahora → Es probable que X se accese en el futuro cercano
tipo
Espacial
es
Si X se accesa ahora → Es probable que datos cercanos a X se accesen también
tipo
∞ Procesadores
tipo
M. Compartida
P_1 P_2 P_{n_p}
Red de Interconexión
M
tipo
M. Distribuida
Red de Interconexión
P_1 P_2 P_{n_p}
M_1 M_2 M_{n_p}
Memoria
Problema
Acceso lento
$t_M \approx 100t_I$
Solución
P_1 P_2 P_{n_p}
M_1^c M_2^c $M_{n_p}^c$
M
M^c
M
Memoria Cache
Red de Interconexión
P_1 P_2 P_{n_p}
M_1^c M_2^c $M_{n_p}^c$
M_1 M_2 M_{n_p}


MEMORIA: Cache
tipo
1 Procesador
tipo
∞ Procesadores
P
Procesador
M^c
Memoria Cache
Memoria Principal
M
estructura
M
Bloque
[n_C − 1] = 3
[n_M − 1] = 15
7 = [n_B − 1]
Palabra
Dirección de Memoria
Número de Bloque
estructura
M^c
Palabra
Bloque
Dirección de Memoria Cache
tipo
M. Compartida
tipo
M. Distribuida
Memoria Cache
P_1
P_2
P_np
M_1^c
M_2^c
M_np^c
M
Memoria
estructura
M_1^c
M_np^c
[n_C − 1] =
Palabra
Bloque
Dirección de Memoria Cache
M
[n_M − 1] =
Dirección de Memoria
Número de Bloque
Red de Interconexión
Memoria Cache
M_1
M_2
M_np
Memoria
estructura
M_1^c
M_np^c
[n_C − 1] = 3
Palabra
Bloque
Dirección de Memoria Cache
M_1
M_2
M_np
Dirección de Memoria
Número de Bloque


MEMORIA: Cache
tipo
1 Procesador
tipo
∞ Procesadores
P
Procesador
M^c
Memoria Cache
M
Memoria Principal
tipo
M. Compartida
M. Distribuida
Red de Interconexión
Memoria Cache
estructura
estructura
tiene
Mapeo
tipo
Asociativo de n-vías
n=1
n=2
n= n_C
usa
n=2,4,8
Directo
Completamente Asociativo
Bloque
Palabra
[n_C − 1] = 3
[n_M − 1] = 15
7 = [n_B − 1]
Dirección de Memoria
Número de Bloque
Dirección de Memoria Cache
es
Define en qué dirección de la memoria caché se guardara una copia del contenido de una dirección de la memoria

MEMORIA: Cache

tipo

1 Procesador

P — Procesador

M^c — Memoria Cache

M — Memoria Principal

estructura

M

0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 — $[n_M - 1] = 15$

0 Bloque
1
2
3
4
5
6
7 = $[n_B - 1]$

Palabra

Dirección de Memoria

Número de Bloque

estructura

M^c

0 1 2 $[n_C - 1] = 3$

Palabra

Bloque

Dirección de Memoria Cache

tiene

Reemplazo

es

Define qué dirección de la memoria caché se va a reemplazar por una nueva cuando la memoria cache esta llena

tipo

FIFO = Primero en Entrar Primero en Salir (First-In-First-Out)

t

2 → 2
0 → 0 2
1 → 1 0 2
3 → 3 1 0 2
2 3 1 0

Cola de Direcciones de Memoria Cache

Dirección de Memoria Cache

tipo

Aleatorio

función

Reemplaza la línea de cache al azar

tipo

LFU = Utilizado con menos Frecuencia (Least Frequently Used)

M^c C

0 1 2 3 — 1 6 2 3

Palabra

Bloque

Se reemplaza la línea de memoria cache con el contador más bajo

Contador de Número de Referencias a la Dirección de Memoria Cache

tipo

LRU = Menos usado Recientemente (Least Recently Used)

usa

Lista de Direcciones de Memoria Cache

3–0–1–2

regla 1

Al hacer referencia a la dirección de memoria cache 1 esta se pasa al frente de la lista

3–0–1–2

regla 2

Se reemplaza el ultimo en la lista y se pasa al frente

1–3–0–2

tipo

∞ Procesadores

tipo

M. Compartida

P_1 P_2 … P_{n_p}

M_1^c M_2^c … $M_{n_p}^c$

M

Memoria Cache

tipo

M. Distribuida

Red de Interconexión

P_1 P_2 … P_{n_p}

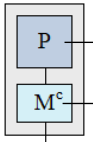
M_1^c M_2^c … $M_{n_p}^c$

M_1 M_2 … M_{n_p}

MEMORIA: Cache

tipo

1 Procesador



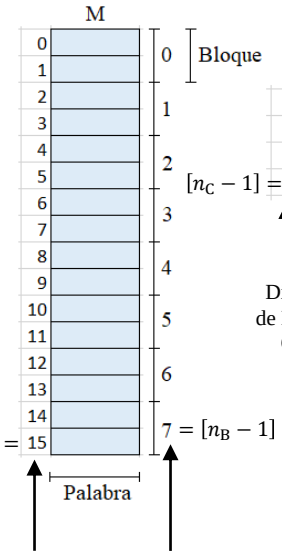
Procesador

Memoria Cache

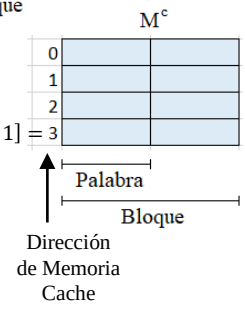
Memoria Principal

M

estructura



estructura

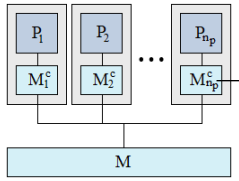


tipo

∞ Procesadores

tipo

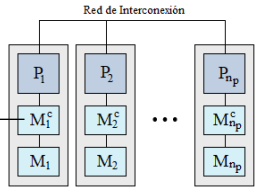
M. Compartida



Memoria Cache

tipo

M. Distribuida



definición

C_H = Coincidencia de Cache (Cache Hit)

Hay una coincidencia de cache si P busca X en M^c y X esta en M^c, es decir, $X \in M^c$

definición

C_M = Fallo de Cache (Cache Miss)

Hay un fallo de cache si P busca X en M^c y X NO esta en M^c, es decir, $X \notin M^c$

definición

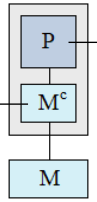
$\uparrow R_{C_H}$ = Razón de Coincidencias de Cache

$$R_{C_H} = \left[\frac{C_H}{C_H + C_M}\right]$$

MEMORIA: Cache

tipo

1 Procesador



Memoria Cache

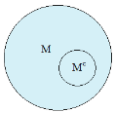
problema

Coherencia de Cache

causa

Redundancia

porque



Los datos en $M^c \subset M$ son una copia de un subconjunto de los datos de M

es

Si X $\in M^c$ y P modifica X en $M^c \rightarrow$ El dato X en M ya no es valido

solución

Actualizar los datos en: M

tipo

Write-throught

Si P modifica X en $M^c \rightarrow$ X se modifica en M también

tipo

Write-back

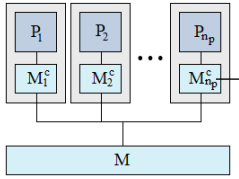
Si P modifica X en $M^c \rightarrow$ X se marca en M^c y cuando se va a borrar X de M^c entonces X se escribe en M

tipo

∞ Procesadores

tipo

M. Compartida



Memoria Cache

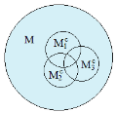
problema

Coherencia de Cache

causa

Redundancia

porque



Los datos en $M_1^c,\ldots, M_{n_p}^c \subset M$ son cada uno 1 copia de un subconjunto diferente de los datos de M

es

Si X $\in M_1^c \cap \cdots \cap M_{n_p}^c$ y P_i modifica X en $M_i^c \rightarrow$ El dato X en $M_1^c, \ldots, M_{(i-1)}^c, M_{(i+1)}^c, \ldots, M_{n_p}^c$ y en M ya no es valido

solución

Actualizar los datos en: M y $M_1^c,\ldots, M_{n_p}^c$

tipo

Write-throught

Si P_i modifica X en $M_i^c \rightarrow$ X se modifica en M también y se invalidan todas las copias de X en $M_{j\neq i}^c$

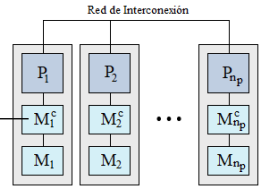
tipo

Write-back

Si P_i modifica X en $M_i^c \rightarrow$ X se marca en M_i^c y se invalidan todas las copias de X en $M_{j\neq i}^c$ y cuando se va a borrar X de M_i^c o $P_{j\neq i}$ quiere leer X entonces se escribe X en M

tipo

M. Distribuida



Memoria Cache

problema

Uso Compartido Falso (False Sharing)

es

Si X, Y $\in M_1^c \cap \cdots \cap M_{n_p}^c$ y X y Y están en el mismo bloque de cache, y P_i solo usa X y P_j solo usa Y $\rightarrow$ Si P_i modifica X en M_i^c el bloque que contiene a Y en M_j^c se invalida en M_j^c aunque Y no haya sido modificado

tipo

Snooping (en M. Compartida)

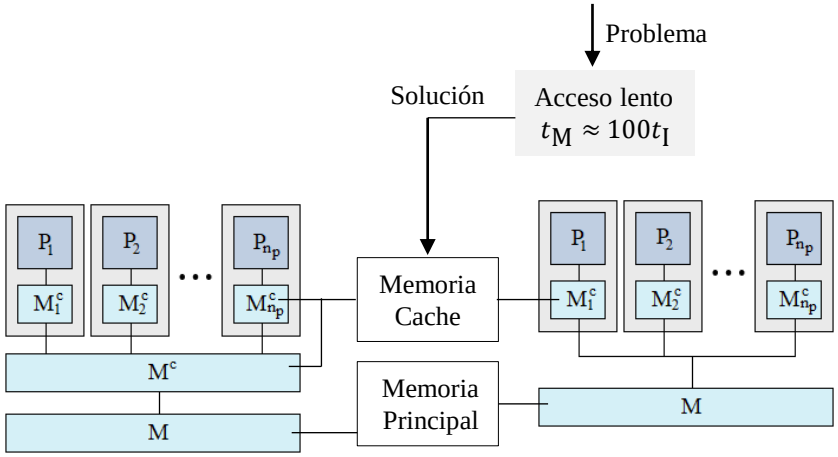
Si P_i modifica el dato X en $M_i^c \rightarrow P_i$ avisa a todos P_j que su copia de X es invalida

tipo

Basado en Directorio (en M. Distribuida)

Hay una estructura de datos D_i que mantiene el estado del dato X en M_i^c

APENDICE: Ecuaciones

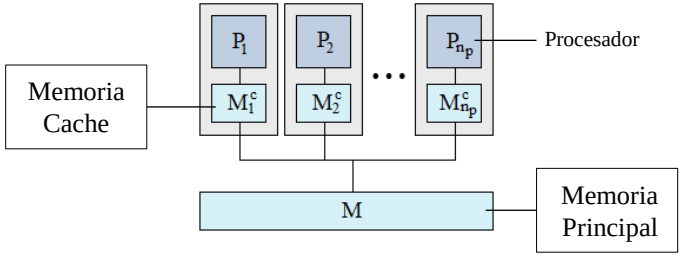


Bus Snooping

3–0–1–2

1–3–0–2

2–3–0–1



Si P_i modifica el dato X en $M_3^c \rightarrow$ P_i avisa a todos P_j que tengan X en M_j^c que su dato X es invalido

$$\left[\frac{(1-a)t_S}{n_p}\right] \quad \left[\frac{1}{n_p}\right](1-a)t_S$$

$T_S \; T_{S\rightarrow P} \quad \alpha a t_S \; \beta(1-a)t_S \quad n \quad n_p n \quad n_p(1-a)t_S =$
Tiempo
en paralelo

$T_S \; T_{S\rightarrow P} \quad \alpha a t_S \; \beta(1-a)t_S \; n \quad n_p(1-a)t_S =$
Tiempo
en paralelo

$$\left[\frac{(1-a)t_S}{n_p}\right] \quad \left[\frac{1}{n_p}\right](1-a)t_S$$