

# **Ingeniería de los Computadores**

Sesión 6. Paralelismo. Conceptos y técnicas

### Introducción

- #1 TOP500 (6/2012) - Sequoia
  - Manufacturer: IBM
  - Núcleos: 1572864
  - OS: Linux
  - Interconexión: a medida
  - Rendimiento máximo: 16324,8 TF
  - Rendimiento pico: 20132,7 TF
  - Consumo: 7890 kW
  - Lugar: Lawrence Livermore National Laboratory (LLNL)
  - <https://computing.llnl.gov/>
  - Especificaciones detalladas:  
[https://computing.llnl.gov/?set=resources&page=OCF\\_resources#sequoia](https://computing.llnl.gov/?set=resources&page=OCF_resources#sequoia)

### Introducción

- #5 TOP500 (6/2012) – Tianhe 1A
  - #1 TOP500 (11/2010)
  - Manufacturer: NUDT (National University of Defense Technology)
  - Núcleos: 186368
  - OS: Linux
  - Interconexión: propietaria
  - Rendimiento máximo: 2566 TF
  - Rendimiento pico: 4701 TF
  - Consumo: 4040 kW
  - Lugar: National Supercomputing Center in Tianjin
  - <http://www.nscg-tj.gov.cn/en/>
  - Especificaciones detalladas: [http://www.nscg-tj.gov.cn/en/resources/resources\\_1.asp](http://www.nscg-tj.gov.cn/en/resources/resources_1.asp)

### Introducción

- #177 TOP500 (6/2012) – Barcelona Supercomputing Center
  - Manufacturer: Bull
  - Núcleos: 5544
  - OS: Linux
  - Interconexión: Infiniband
  - Rendimiento máximo: 103,2 TF
  - Rendimiento pico: 182,9 TF
  - Consumo: 81,50 kW
  - Lugar: Barcelona Supercomputing Center
  - <http://www.bsc.es/>
  - Especificaciones detalladas: <http://www.bsc.es/marenostrum-support-services/res>

### Introducción

- Y todo esto...¿para qué?
  - Defensa
    - Terrorismo
    - Programa armamentístico
    - Programa espacial
  - Energía
    - Combustible basado en hidrógeno
    - Prospección de energías (carbón, gas, petróleo)
  - Ingeniería
    - Nanotecnología
    - Modelado biológico
    - Análisis de imagen

- Uso de un supercomputador

- Procesamiento paralelo

Estudia los aspectos relacionados con la división de una aplicación en unidades independientes y su ejecución en múltiples procesadores para reducir tiempo y/o aumentar la complejidad del problema.

- Procesamiento distribuido

Estudia los aspectos relacionados con la ejecución de múltiples aplicaciones al mismo tiempo utilizando múltiples recursos (procesadores, memorias, discos y bases de datos) situados en distintas localizaciones físicas.

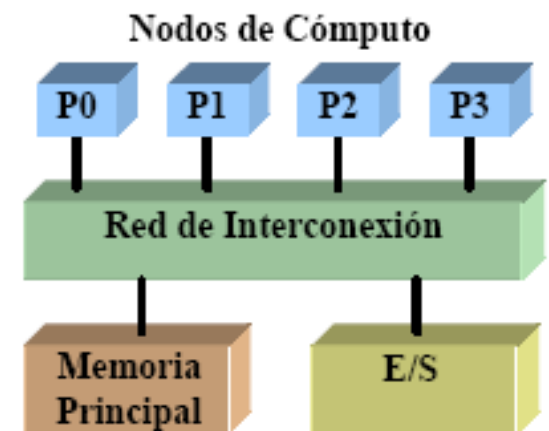
# Ingeniería de los Computadores

## Sesión 6. Paralelismo

### Introducción

### Conceptos

- Clasificación de computadores paralelos (según el sistema de memoria)
  - Multiprocesadores. Comparten el mismo espacio de memoria (el programador no necesita saber dónde están los datos)
  - Multicomputadores. Cada procesador (nodo) tiene su propio espacio de direcciones (el programador necesita saber dónde están los datos)
- Diseño de un computador paralelo
  - Nodos de cómputo
  - Sistema de memoria
  - Sistema de comunicación
  - Sistema de entrada/salida



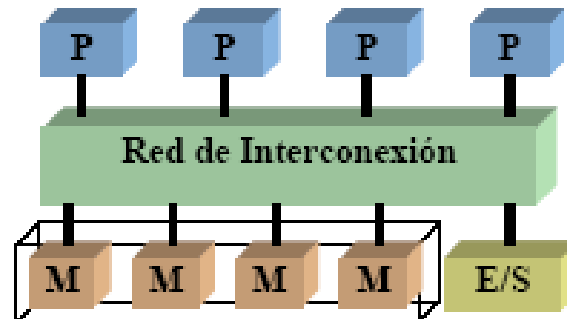
# Ingeniería de los Computadores

## Sesión 6. Paralelismo

Introducción

Conceptos

- Multiprocesador con memoria centralizada (SMP – Symmetric MultiProcessor)
  - Mayor latencia
  - Poco escalable
  - Comunicación mediante variables compartidas (datos no duplicados en memoria principal)
  - Necesita implementar primitivas de sincronización
  - No necesita distribuir código y datos
  - Programación más sencilla (generalmente)





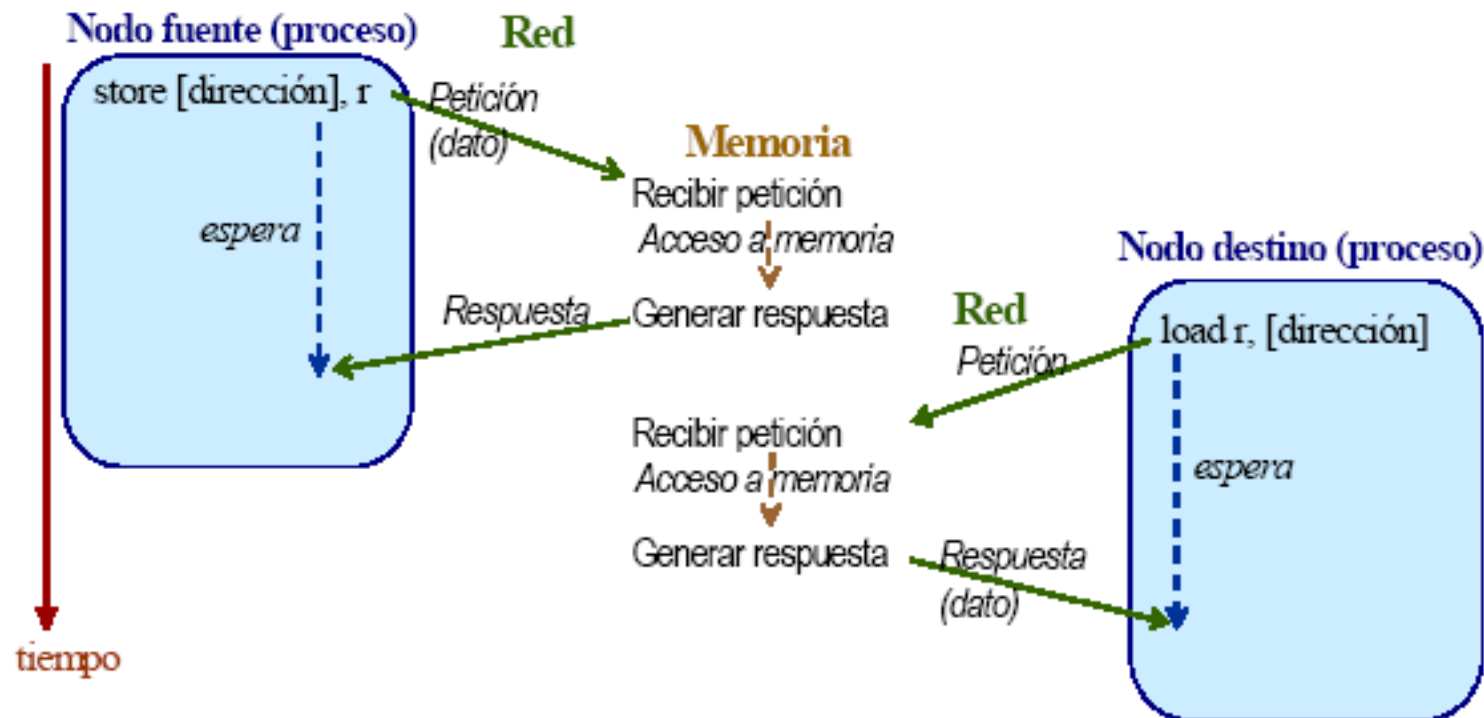
# Ingeniería de los Computadores

## Sesión 6. Paralelismo

Introducción

Conceptos

- Comunicación en un multiprocesador



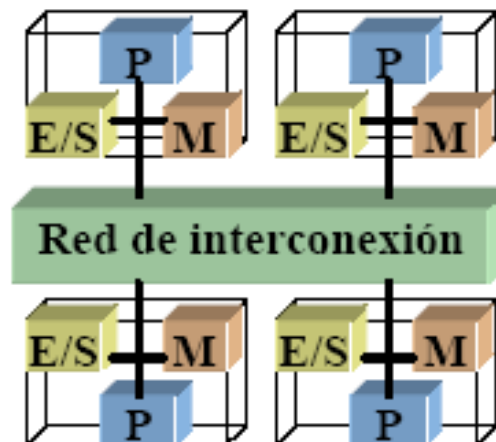
# Ingeniería de los Computadores

## Sesión 6. Paralelismo

Introducción

Conceptos

- Multicomputador
  - Menor latencia
  - Mayor escalabilidad
  - Comunicación mediante paso de mensajes (datos duplicados en memoria)
  - Sincronización mediante mecanismos de comunicación
  - Distribución de carga de trabajo (código y datos) entre procesadores
  - Programación más difícil



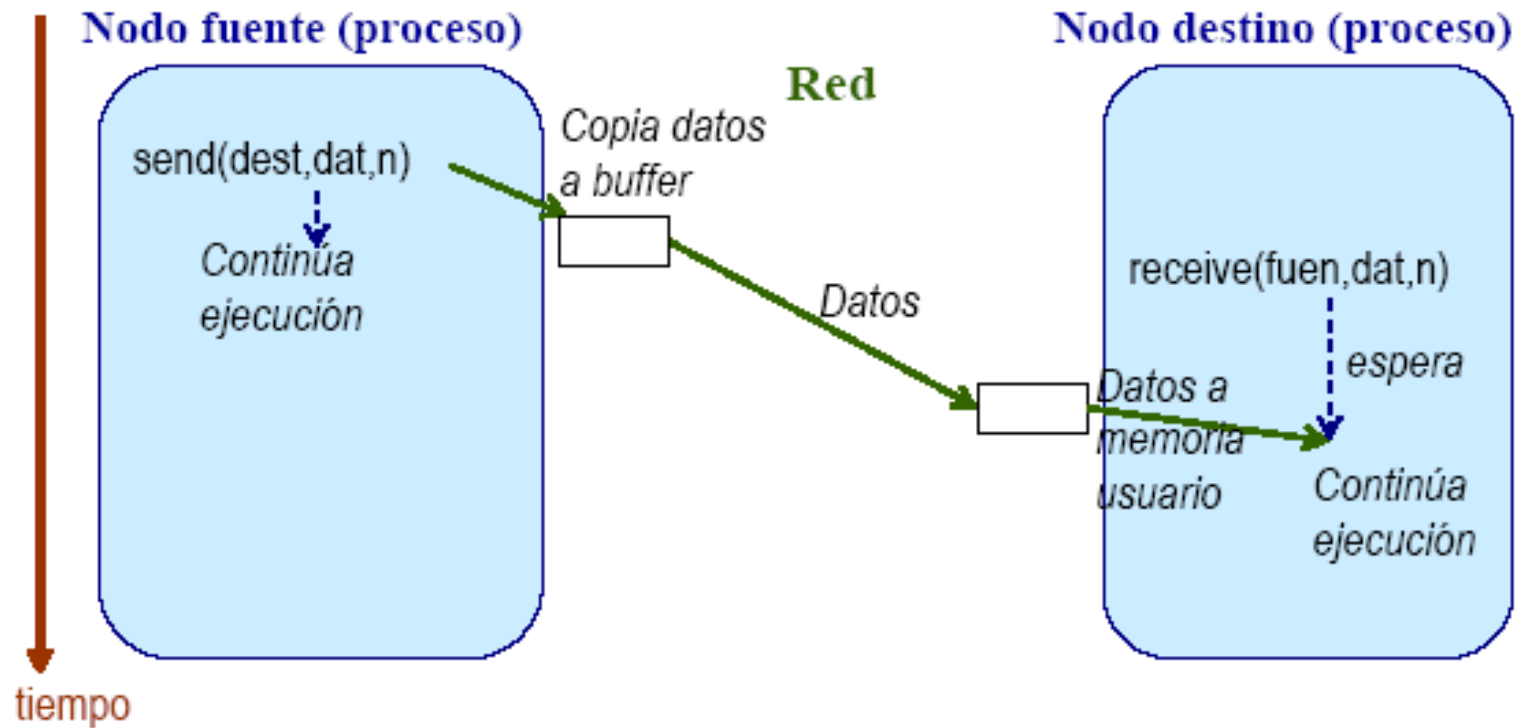
# Ingeniería de los Computadores

## Sesión 6. Paralelismo

Introducción

Conceptos

- Comunicación asíncrona en un multicomputador



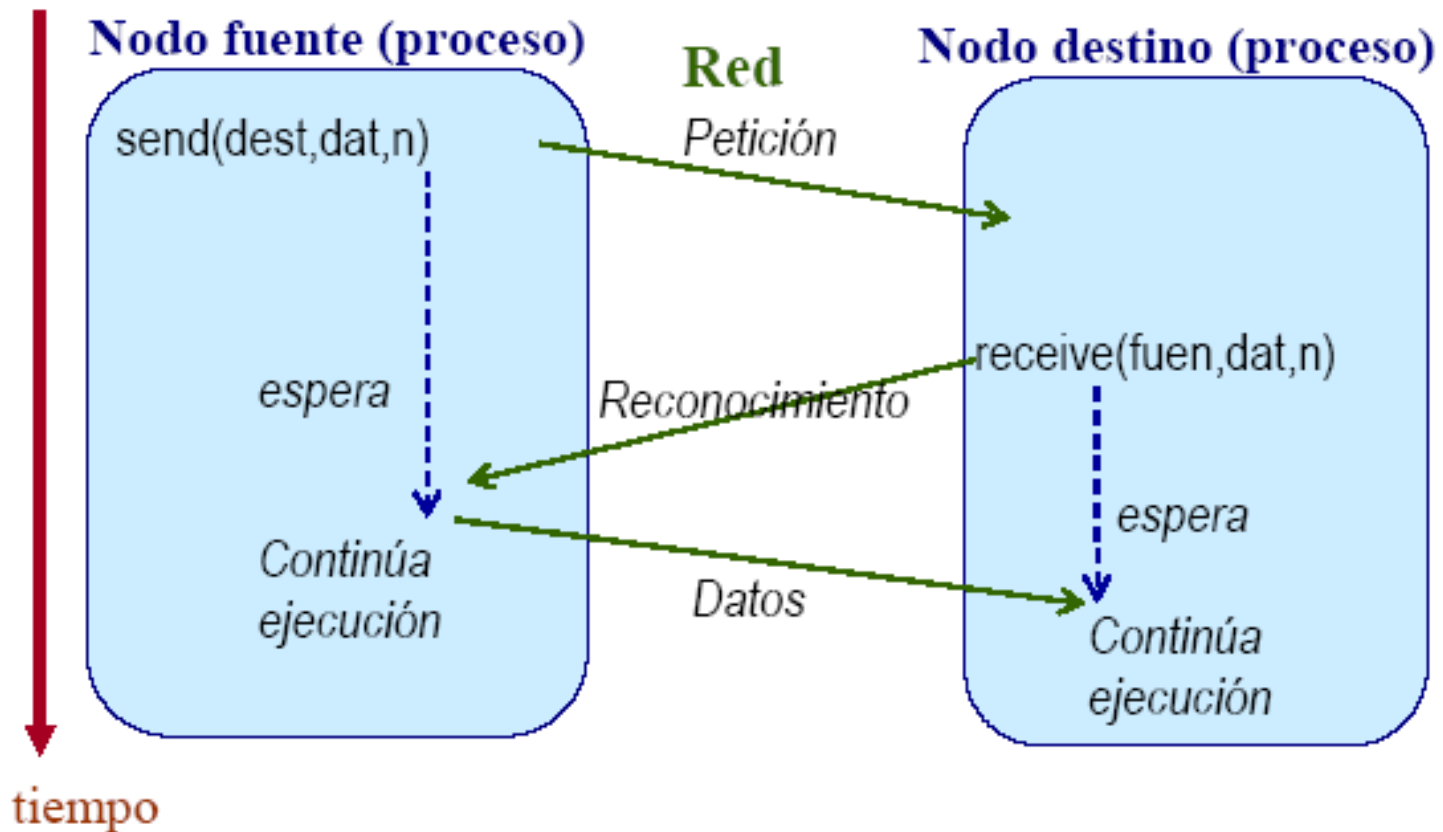
# Ingeniería de los Computadores

## Sesión 6. Paralelismo

Introducción

Conceptos

- Comunicación síncrona en un multicomputador



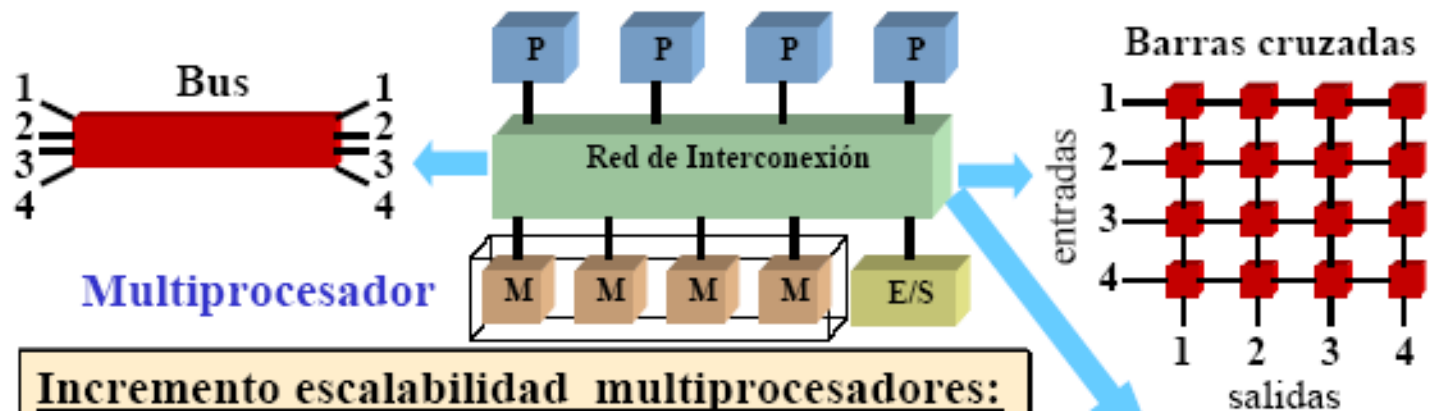
# Ingeniería de los Computadores

## Sesión 6. Paralelismo

### Introducción

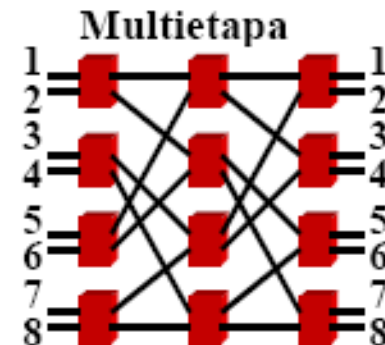
### Conceptos

- Redes de interconexión en multiprocesadores



### Incremento escalabilidad multiprocesadores:

- Añadir **cache local** a cada procesador.
- Usar **redes de menor latencia y mayor ancho de banda** que un bus (jerarquía de buses, barras cruzadas, multietapa).
- **Distribuir físicamente los módulos de memoria** entre los procesadores (pero se sigue compartiendo espacio de direcciones).



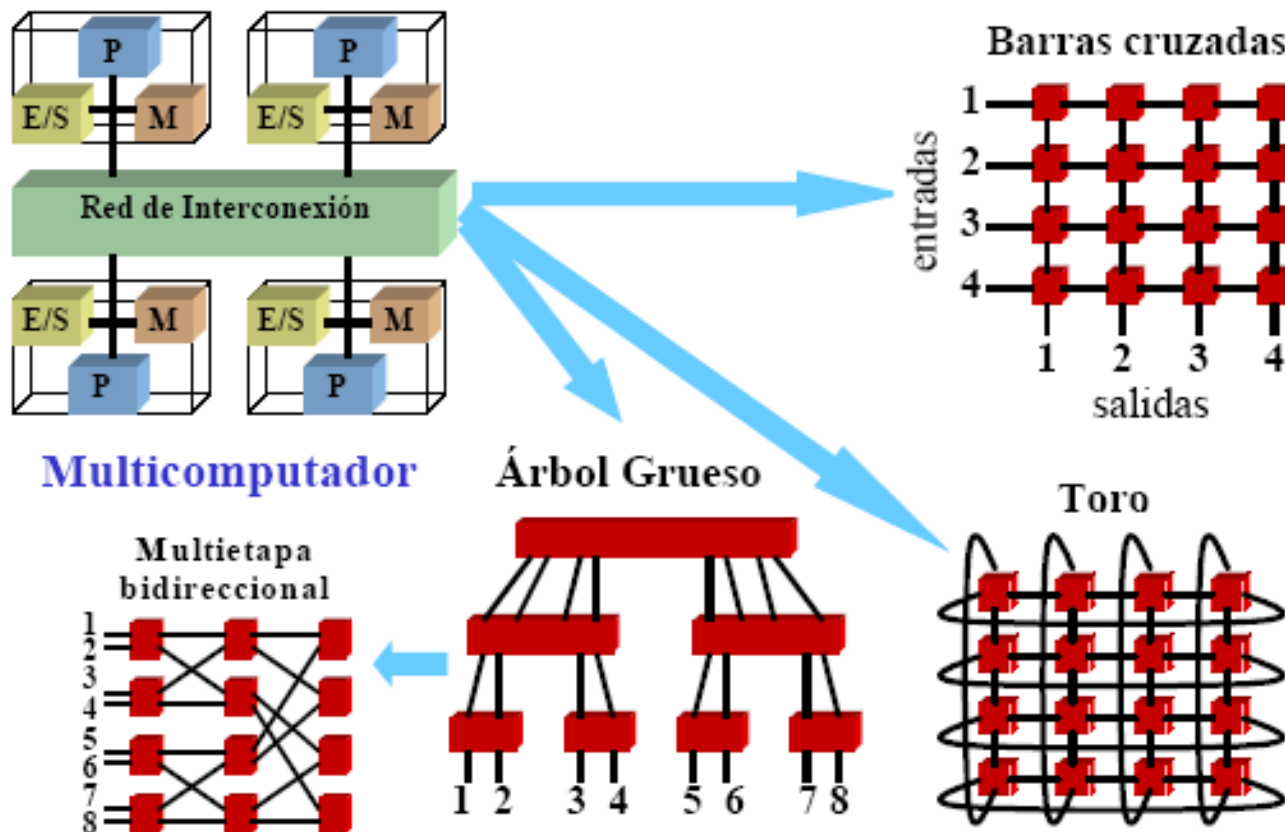
# Ingeniería de los Computadores

## Sesión 6. Paralelismo

Introducción

Conceptos

- Redes de interconexión en multicomputadores



# Ingeniería de los Computadores

## Sesión 6. Paralelismo

Introducción

Conceptos

- Clasificación de computadores paralelos

<b>Multicomputadores</b> Memoria no compartida Múltiples espacios de direcciones.	IBM-SP2 HP AlphaServer SC45 (cluster de SMP)			Memoria <b>fisicamente distribuida</b>          <b>Escalables</b>	+          Escalabilidad          1
<b>Multiprocesadores</b> Memoria compartida Un único espacio de direcciones.	<b>NUMA</b> <i>(Non-Uniform Memory Access)</i>	<b>NUMA</b>	Cray T3E, Cray X1		
		<b>CC-NUMA</b>	Origin 3000 HP 9000 Superdome		
		<b>COMA</b>	KSR-1		
	<b>UMA</b> <i>(Uniform Memory Access)</i>	<b>SMP</b>	WS, servidores básicos	Memoria <b>fisicamente centralizada</b>	

- Otros tipos de computadores paralelos
  - **MPP (Massive Parallel Processor)**. Número de procesadores superior a 100. Sistemas con redes diseñadas a medida.
  - **Cluster**. Computadores completos (PC's, servidores, etc.) conectados con alguna red comercial tipo LAN que se utiliza como recurso de cómputo único. El tráfico de la red se reduce al ocasionado por la aplicación ejecutada (no hay tráfico externo)
  - **Cluster Beowulf**. Cluster con sistema operativo libre (Linux) y hardware y software de amplia difusión
  - **Constelaciones**. Cluster de SMP con un  $n^o$  de procesadores en un nodo mayor que el  $n^o$  de nodos del cluster



- Otros tipos de computadores paralelos
  - **Redes de computadores.** Conjunto de computadores conectados por una LAN. Por la red circula tanto tráfico de la aplicación paralela como externo
  - **GRID.** Conjunto de recursos autónomos distribuidos geográficamente conectados por infraestructura de telecomunicaciones y que conforman un sistema de altas prestaciones virtual

- Tipos de paralelismo
  - Paralelismo funcional

Se obtiene a través de la reorganización de la estructura lógica de una aplicación. Existen diferentes niveles de paralelismo funcional según las estructuras que se reorganicen.

- Paralelismo de datos

Implícito en operaciones con estructuras de datos tipo vector o matriz. Está relacionado con operaciones realizadas sobre grandes volúmenes de datos que sean independientes entre si.

# Ingeniería de los Computadores

## Sesión 6. Paralelismo

Introducción

Conceptos

- Paralelismo funcional

Niveles:

**Programas**

Programa1

Programa2

.....

**Funciones**

```
Func1() {  
...  
}
```

```
Func2() {  
...  
}
```

```
Func3() {  
...  
}
```

**Bucle  
(bloques)**

```
for ( ) {  
...  
}
```

```
while( ) {  
...  
}
```

**Operaciones**

\*

+

/

Granularidad:

**Grano Grueso**

**Grano  
Medio**

**Grano  
Medio-Fino**

**Grano Fino**

- Tipos de paralelismo

- Paralelismo explícito

Paralelismo no presente de forma inherente en las estructuras de programación y que se debe indicar expresamente.

- Paralelismo implícito

Paralelismo presente (aunque puede no estar ejecutándose de forma paralela) debido a la propia estructura de los datos (vectores) o de la aplicación (bucle)

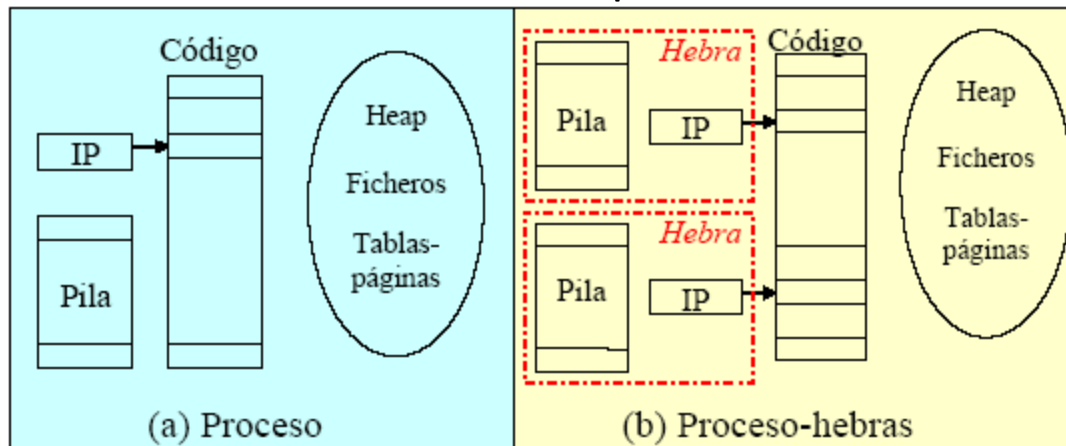
# Ingeniería de los Computadores

## Sesión 6. Paralelismo

### Introducción

### Conceptos

- Unidades de ejecución: hilos y procesos
  - Hardware (procesador). Gestiona la ejecución de las instrucciones
  - Software (SO). Gestiona la ejecución de hilos y procesos
    - Proceso: espacio de direcciones virtuales propio
    - Hilos: comparten direcciones virtuales, se crean y destruyen más rápido y la comunicación también es más rápida

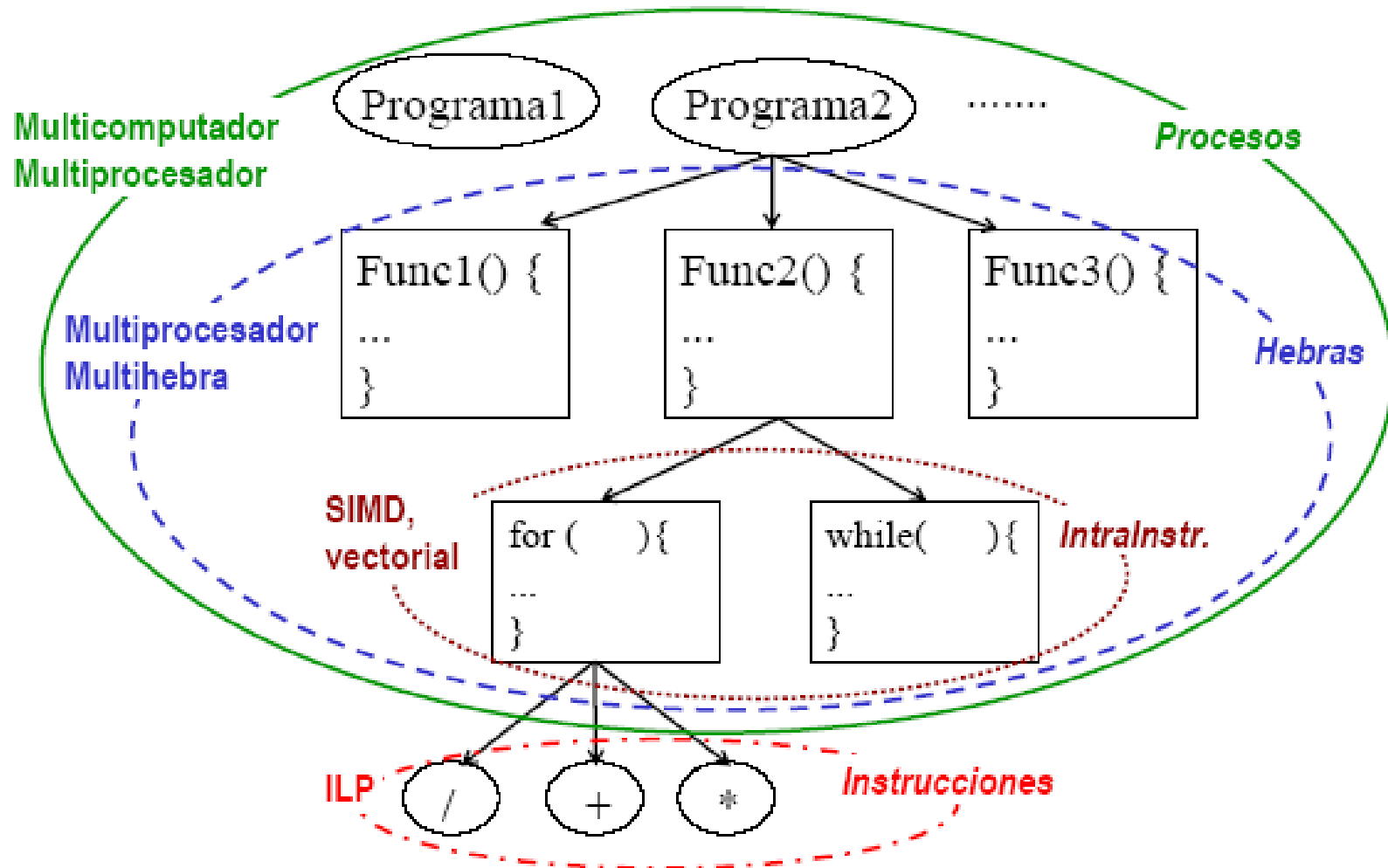


# Ingeniería de los Computadores

## Sesión 6. Paralelismo

Introducción

Conceptos



# Ingeniería de los Computadores

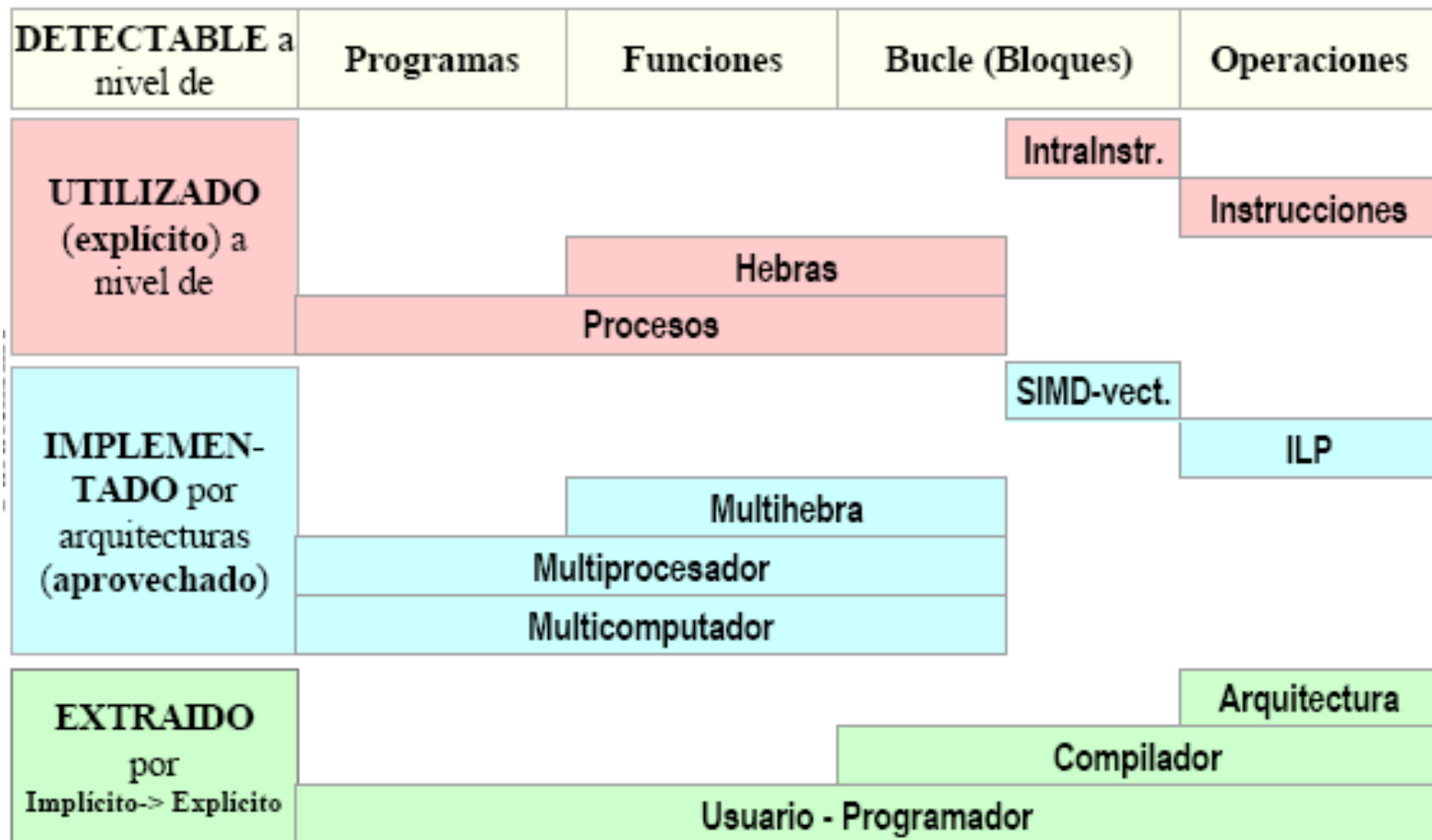
## Sesión 6. Paralelismo

Introducción

Conceptos

Prog. Paralela

- Detección y extracción de paralelismo



# Ingeniería de los Computadores

## Sesión 6. Paralelismo

Introducción

Conceptos

Prog. Paralela

- Problemas introducidos por la programación paralela
  - División en unidades de cómputo independientes (tareas)
  - Agrupación de tareas (código y datos) en procesos/hebras
  - Asignación a procesadores
  - Sincronización y comunicación
- Situación inicial (normalmente)
  - Se parte de una versión secuencial (no paralela)
  - Se parte de una descripción de la aplicación
  - Elementos de apoyo:
    - Programa paralelo que resuelva un problema semejante
    - Librerías de funciones paralelas (BLAS, LAPACK, OpenMP)



# Ingeniería de los Computadores

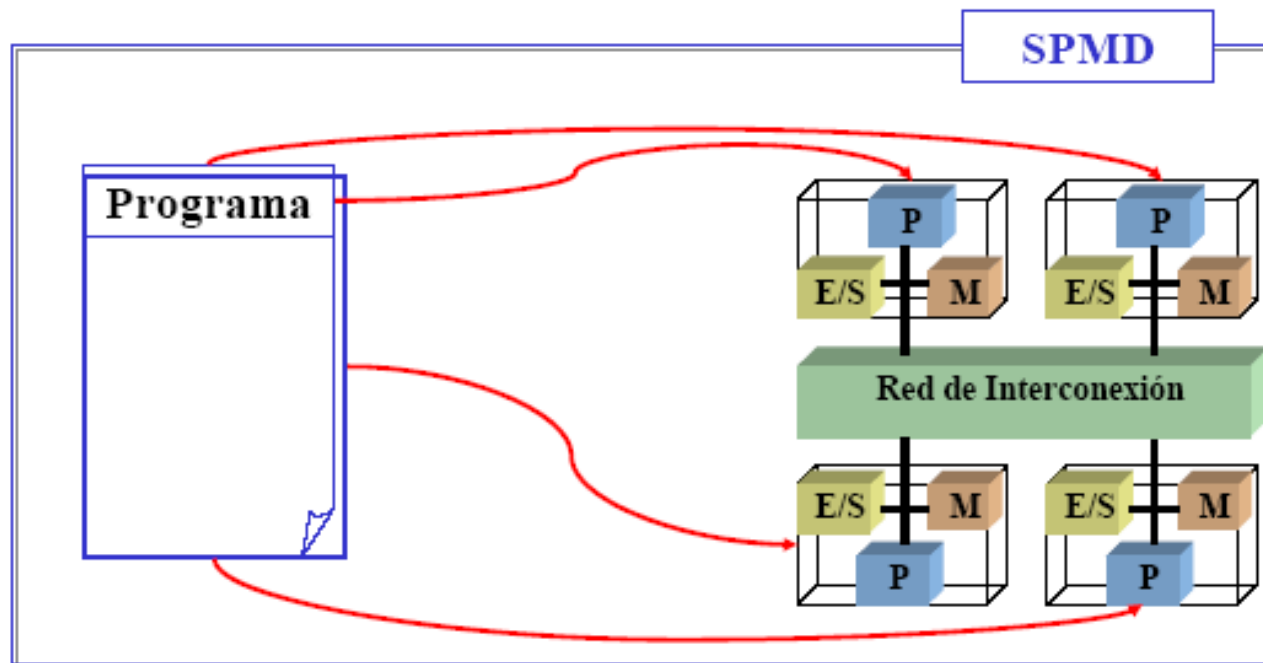
## Sesión 6. Paralelismo

Introducción

Conceptos

Prog. Paralela

- Modos de programación paralela
  - SPMD (Single Program Multiple Data)



# Ingeniería de los Computadores

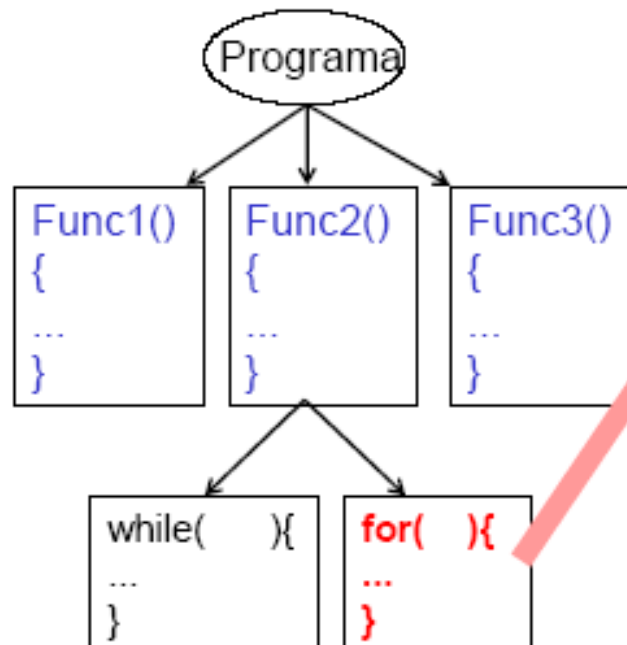
## Sesión 6. Paralelismo

Introducción

Conceptos

Prog. Paralela

- Modos de programación paralela
  - SPMD (Single Program Multiple Data). Ejemplo



**SPMD**

```
Func1() {  
...}  
Func2() {  
...  
for (i=ithread;i<N;i=i+nthread){  
    código para la iteración i  
...}  
Func3() {  
...}  
Main () {  
...  
switch (iproc) {  
    case 0: Func1(); break;  
    case 1: Func2(); break;  
    case 2: Func3(); break; }  
...}
```

# Ingeniería de los Computadores

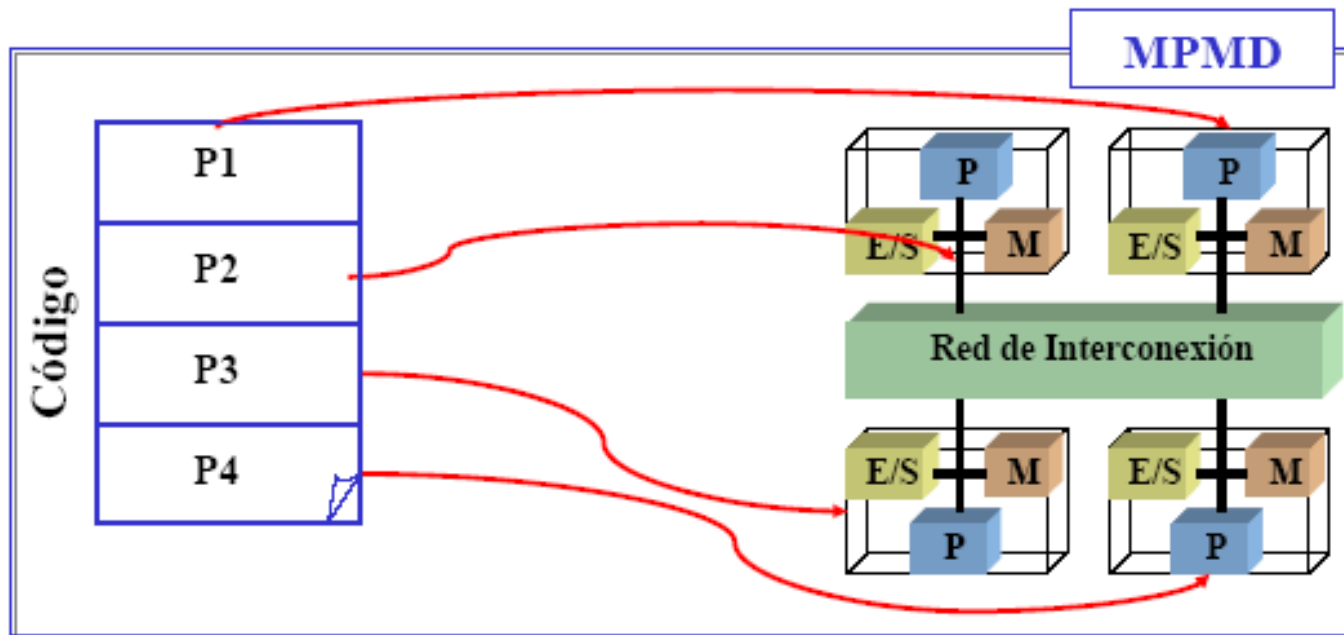
## Sesión 6. Paralelismo

Introducción

Conceptos

Prog. Paralela

- Modos de programación paralela
  - MPMD (Multiple Program Multiple Data)



- Modo Mixto SPMD-MPMD

# Ingeniería de los Computadores

## Sesión 6. Paralelismo

Introducción

Conceptos

Prog. Paralela

- Herramientas que facilitan la programación paralela
  - Compiladores paralelos. Extracción automática del paralelismo
  - Directivas del compilador (OpenMP): lenguaje secuencial + directivas
  - Lenguajes paralelos (HPF, Occam, Ada)
  - Bibliotecas de funciones (Pthread, MPI, PVM): lenguaje secuencial + funciones de biblioteca como interfaces

# Ingeniería de los Computadores

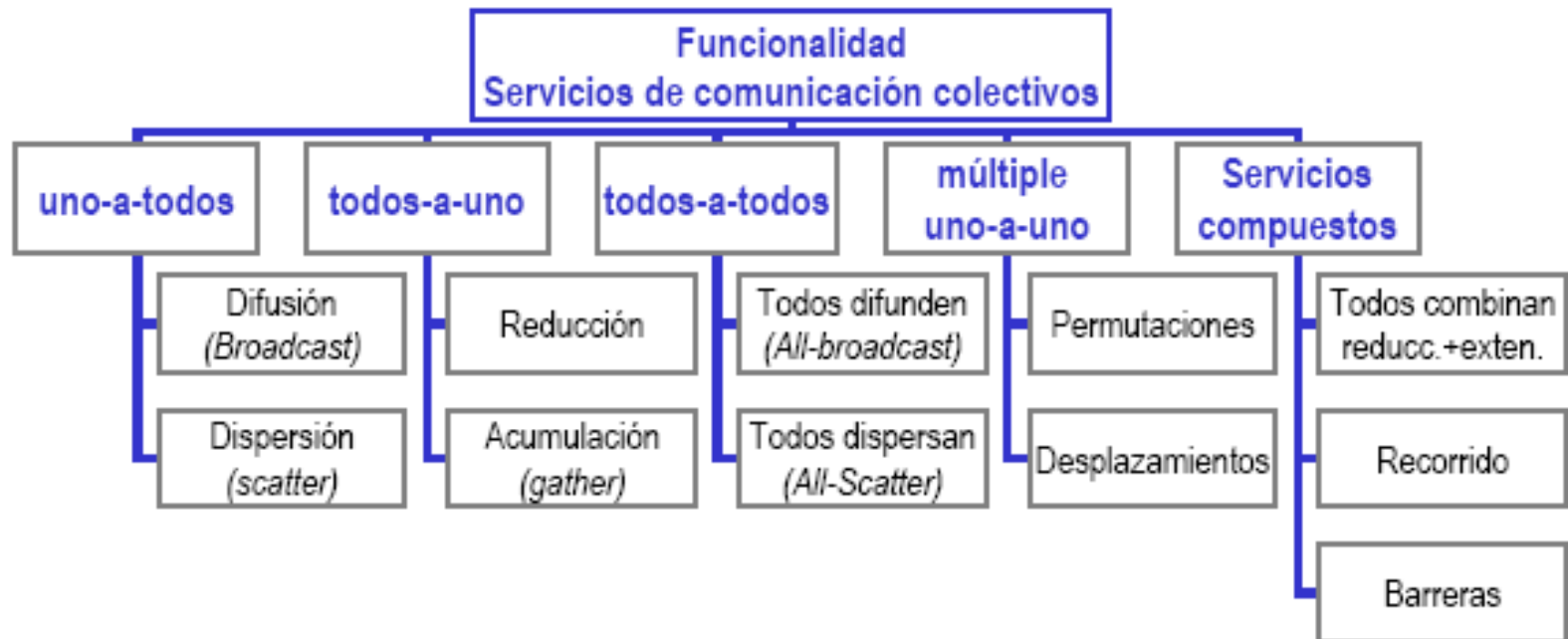
## Sesión 6. Paralelismo

Introducción

Conceptos

Prog. Paralela

- Comunicación. Alternativas



# Ingeniería de los Computadores

## Sesión 6. Paralelismo

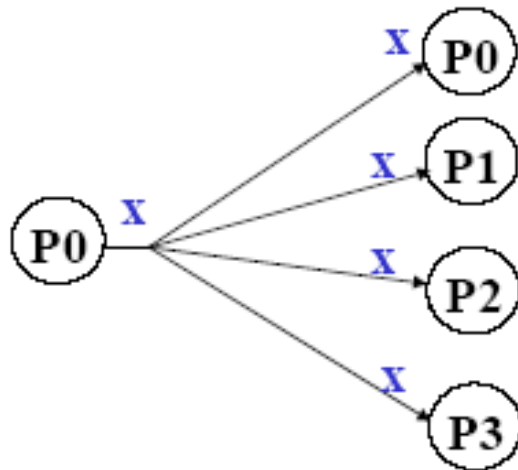
Introducción

Conceptos

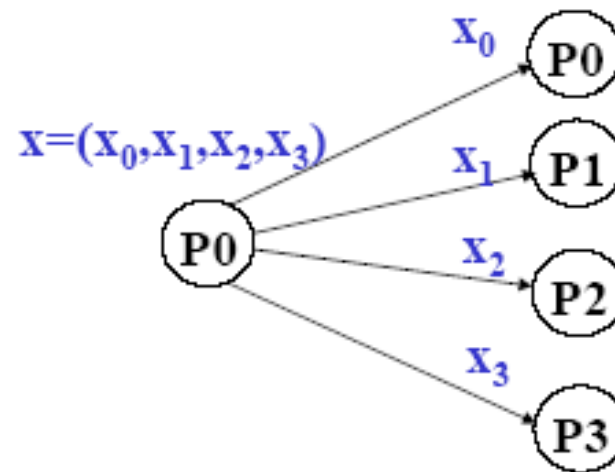
Prog. Paralela

- Comunicación: uno a todos

Difusión (*broadcast*)



Dispersión (*scatter*)



# Ingeniería de los Computadores

## Sesión 6. Paralelismo

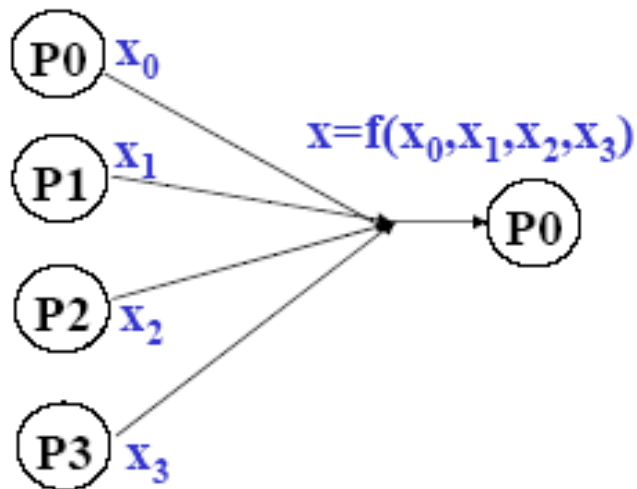
Introducción

Conceptos

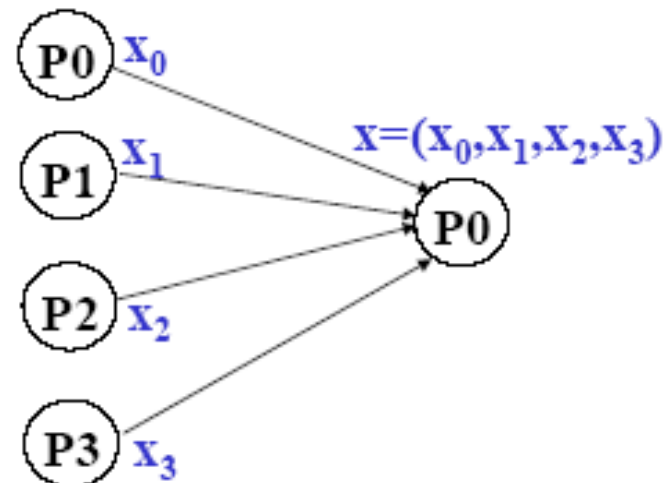
Prog. Paralela

- Comunicación: todos a uno

Reducción



Acumulación (*gather*)



# Ingeniería de los Computadores

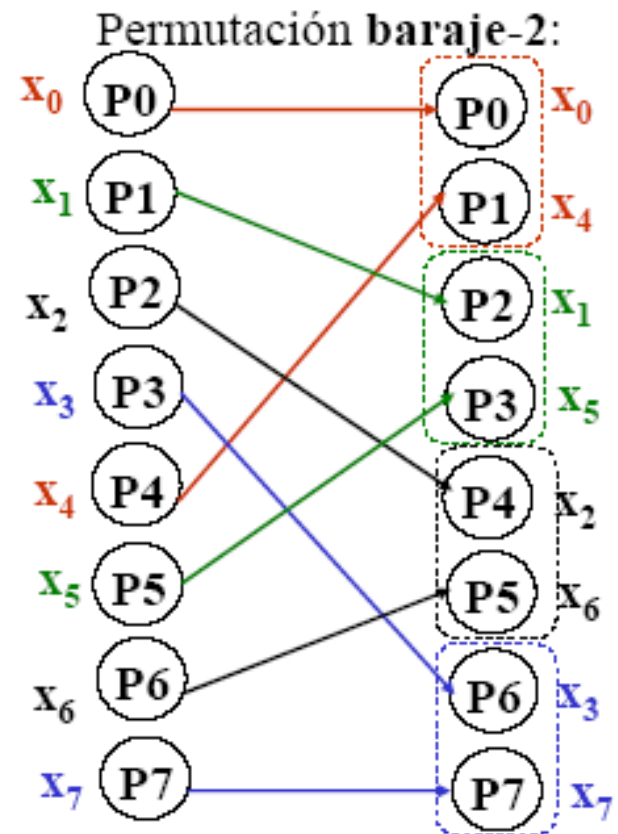
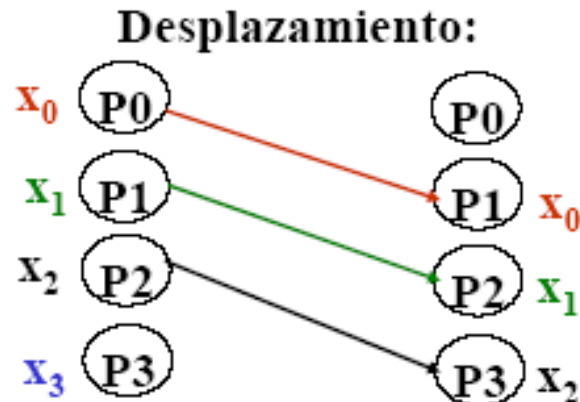
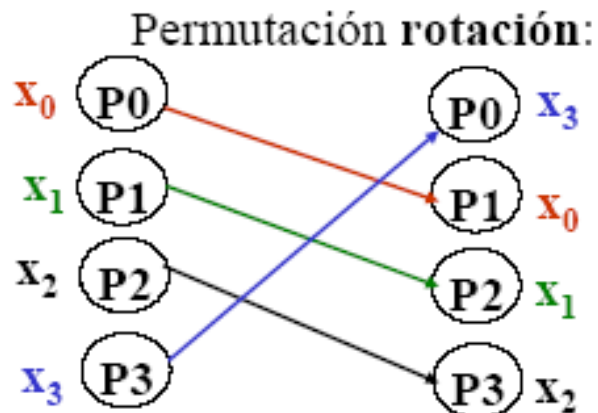
## Sesión 6. Paralelismo

Introducción

Conceptos

Prog. Paralela

- Comunicación: múltiple uno a uno





# Ingeniería de los Computadores

## Sesión 6. Paralelismo

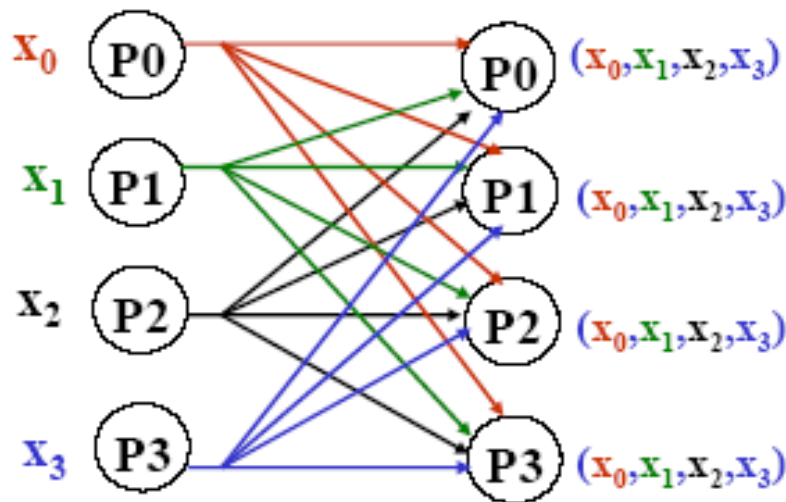
Introducción

Conceptos

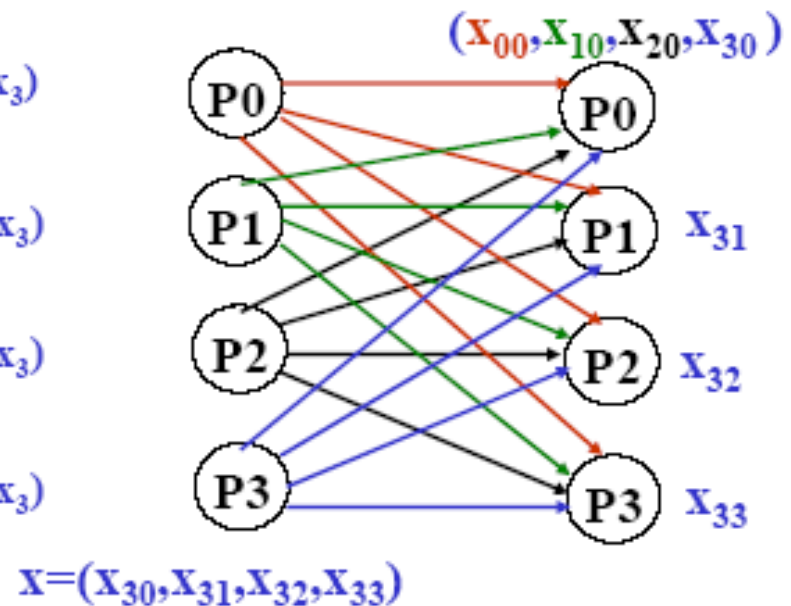
Prog. Paralela

- Comunicación: todos a todos

Todos Difunden (*all-broadcast*)  
o chismorreo (*gossiping*)



Todos Dispersan (*all-scatter*)



# Ingeniería de los Computadores

## Sesión 6. Paralelismo

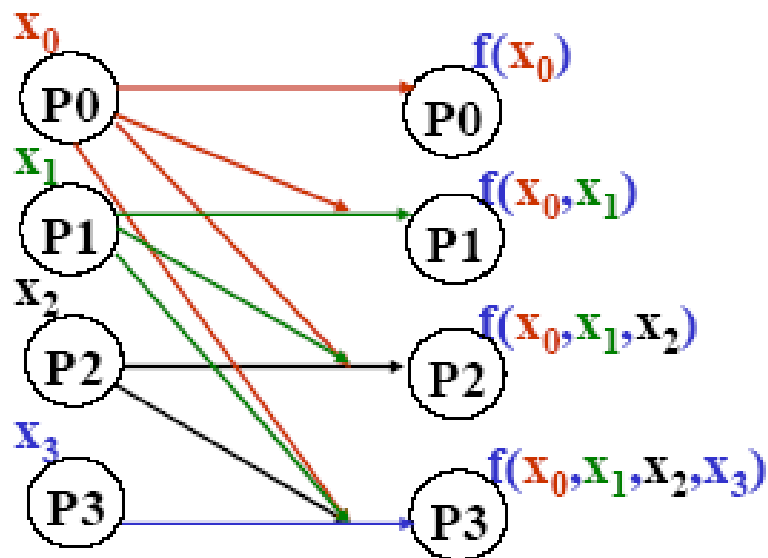
Introducción

Conceptos

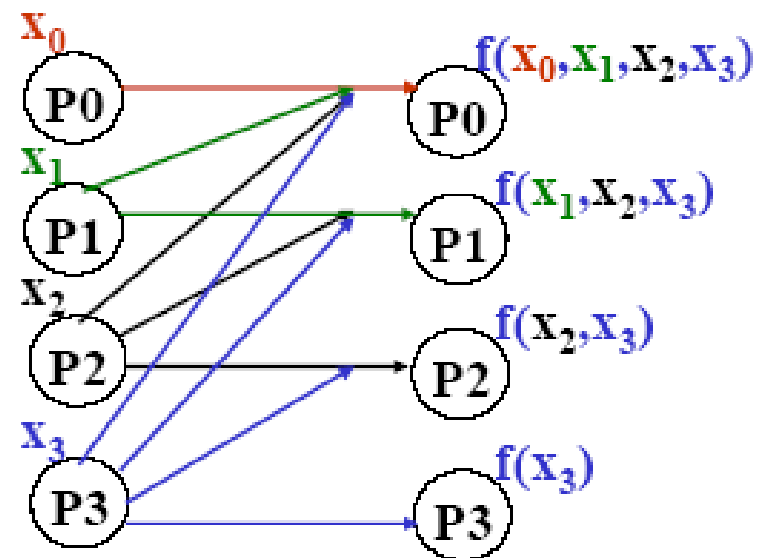
Prog. Paralela

- Comunicación: servicios compuestos

Recorrido prefijo paralelo



Recorrido sufijo paralelo



# Ingeniería de los Computadores

## Sesión 6. Paralelismo

Introducción

Conceptos

Prog. Paralela

- Comunicación. Ejemplos de funciones colectivas usando MPI

<b>Uno-a-todos</b>	<b>Difusión</b>	MPI_Bcast()
	<b>Dispersión</b>	MPI_Scatter()
<b>Todos-a-uno</b>	<b>Reducción</b>	MPI_Reduce()
	<b>Acumulación</b>	MPI_Gather()
<b>Todos-a-todos</b>	<b>Todos difunden</b>	MPI_Allgather()
<b>Servicios compuestos</b>	<b>Todos combinan</b>	MPI_Allreduce()
	<b>Barreras</b>	MPI_Barrier()
	<b>Scan</b>	MPI_Scan

# Ingeniería de los Computadores

## Sesión 6. Paralelismo

Introducción

Conceptos

Prog. Paralela

- Estilos de programación paralela: paso de mensajes
  - Herramientas software
    - Lenguajes de programación (Ada) y librerías (MPI, PVM)
  - Distribución de carga de trabajo
    - Uso de librerías: explícita con construcciones del lenguaje secuencial (if, for, ...)
    - Uso de lenguaje paralelo: explícita con construcciones del propio lenguaje (Occam: sentencia par)
  - Primitivas básicas de comunicación : Send y Receive
  - Funciones de comunicación colectivas
  - Sincronización
    - Receive bloqueante. Barreras (MPI\_Barrier)
    - Send-receive bloqueante en ADA

- Estilo de programación paralela: variables compartidas
  - Lenguajes de programación (Ada), Funciones de librerías (OpenMP), directivas del compilador (OpenMP)
  - Distribución de la carga
    - Librerías: explícita con el lenguaje secuencial
    - Directivas: sincronización implícita (mayor nivel de abstracción)
    - Lenguajes: construcciones del lenguaje
  - Comunicación básica: load y store
  - Funciones de comunicación colectiva
  - Sincronización
    - Semáforos, cerrojos, variables condicionales

# Ingeniería de los Computadores

## Sesión 6. Paralelismo

Introducción

Conceptos

Prog. Paralela

- Estilos de programación paralela: paralelismo de datos
  - Herramientas
    - Lenguaje de programación (HPF – High Performance Fortran)
  - Distribución de carga
    - Directivas para distribuir datos entre procesadores
    - Directivas para paralelizar bucles
  - Comunicación básica: implícita ( $A(i)=A(i-1)$ )
  - Funciones de comunicación colectivas
    - Ímplicitas en funciones usadas explícitamente por el programador (rotaciones – CSHIFT)
  - Sincronización: implícita

# Ingeniería de los Computadores

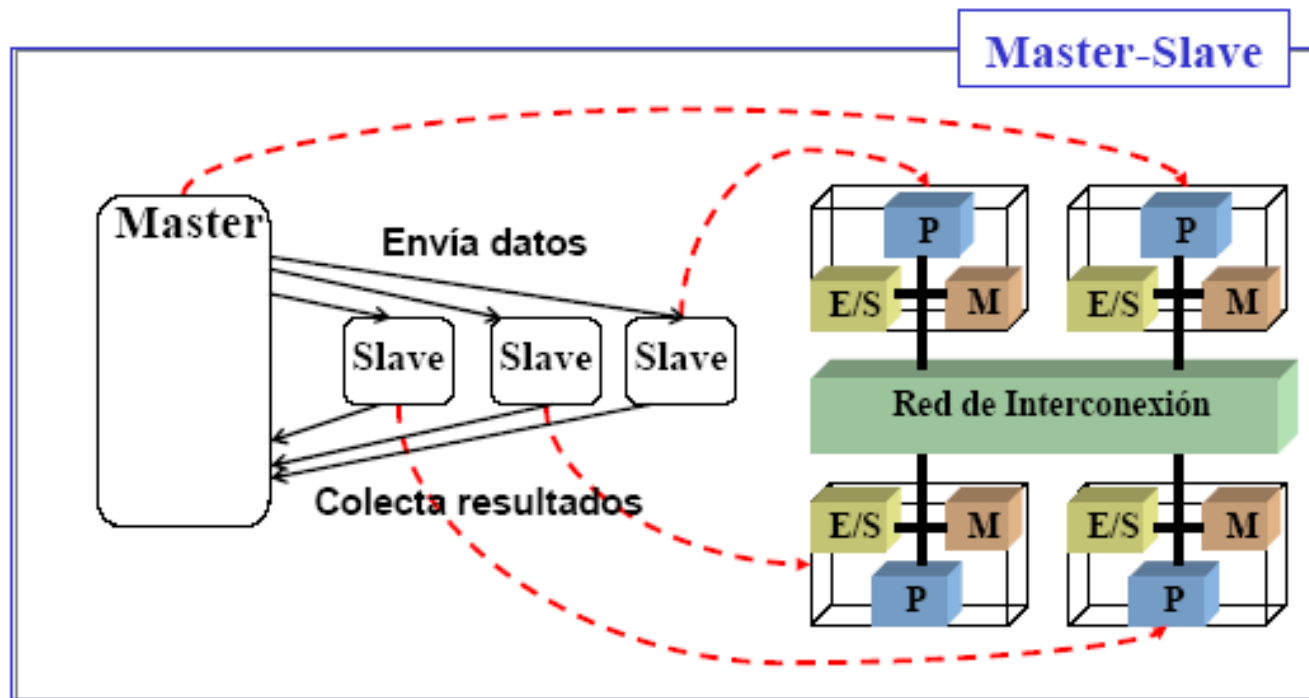
## Sesión 6. Paralelismo

Introducción

Conceptos

Prog. Paralela

- Estructuras de paralelismo: master-slave



# Ingeniería de los Computadores

## Sesión 6. Paralelismo

Introducción

Conceptos

Prog. Paralela

- Estructuras de paralelismo: master slave

### Master-Slave como MPMD– SPMD

```
main ()  
{ código Master  
}  
-----  
main ()  
{ código Slaves  
}
```

### Master-Slave como SPMD

```
main ()  
{ if (iproc=id_Master) {  
    código Master  
} else {  
    código Slaves  
}  
}
```



# Ingeniería de los Computadores

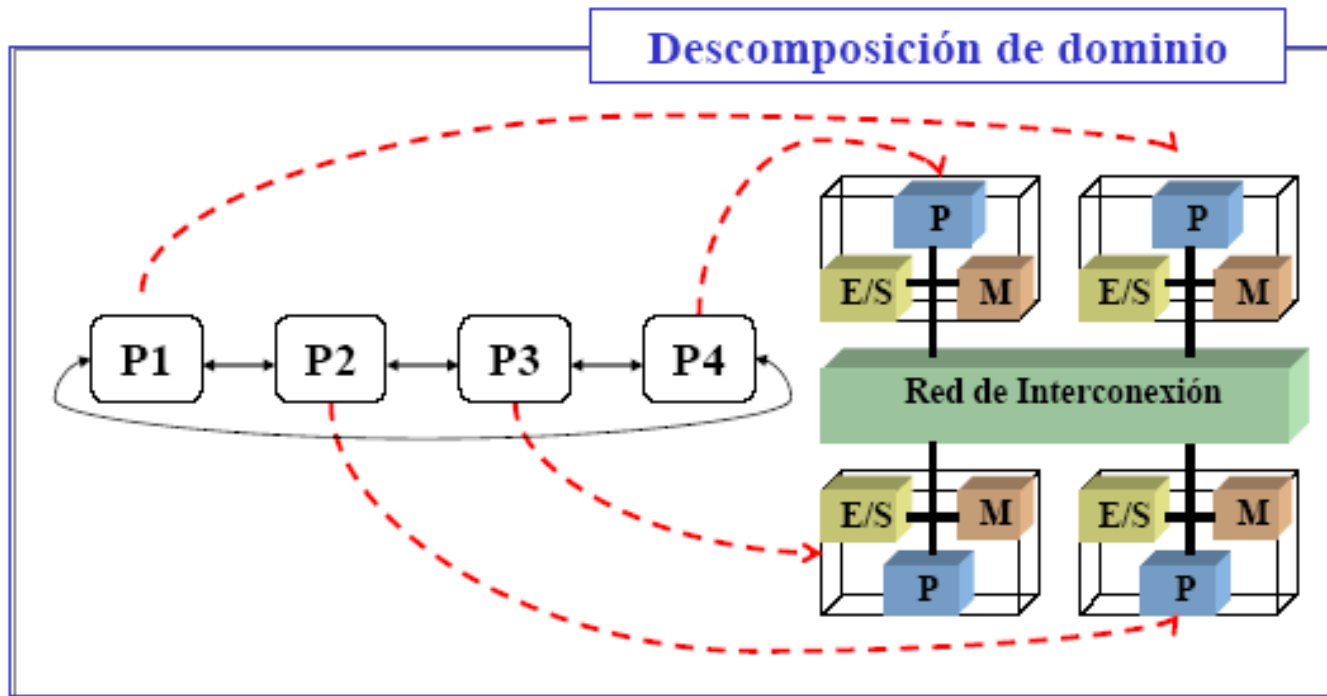
## Sesión 6. Paralelismo

Introducción

Conceptos

Prog. Paralela

- Estructuras de paralelismo: descomposición de dominio (paralelismo de datos)



# Ingeniería de los Computadores

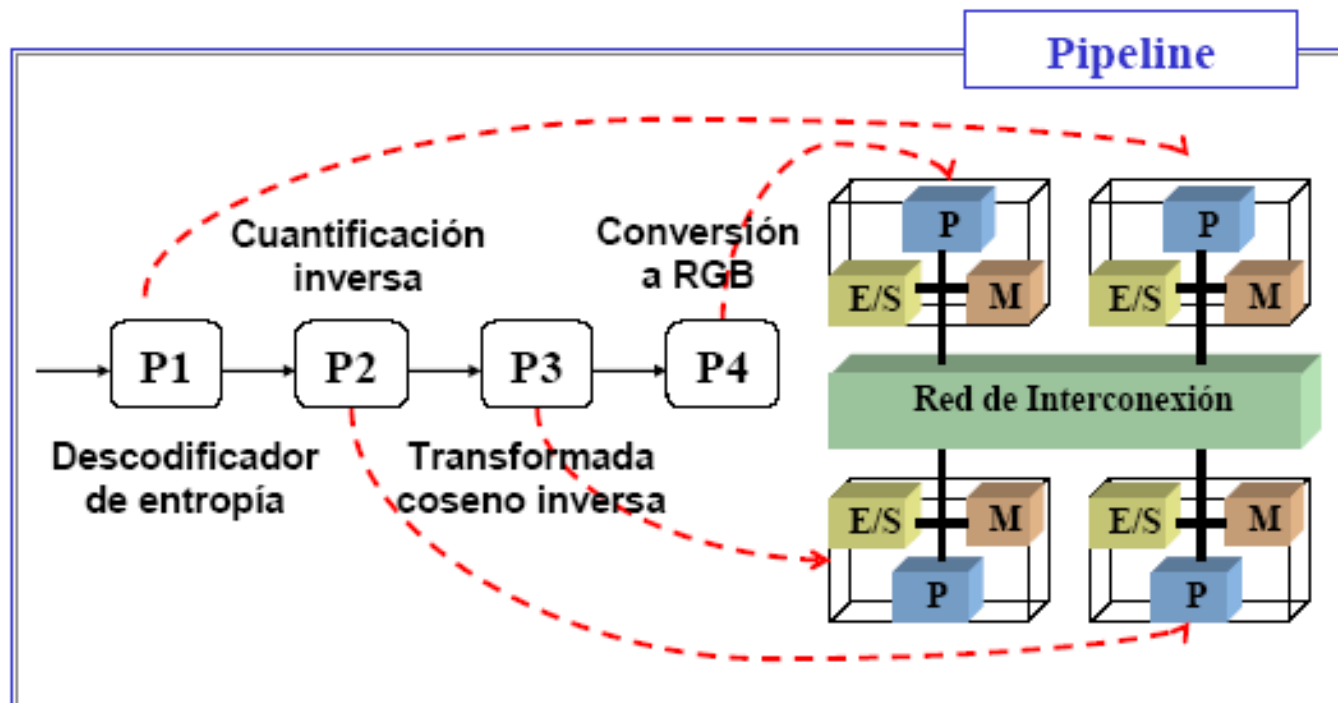
## Sesión 6. Paralelismo

Introducción

Conceptos

Prog. Paralela

- Estructuras de paralelismo: segmentada (flujo de datos)



# Ingeniería de los Computadores

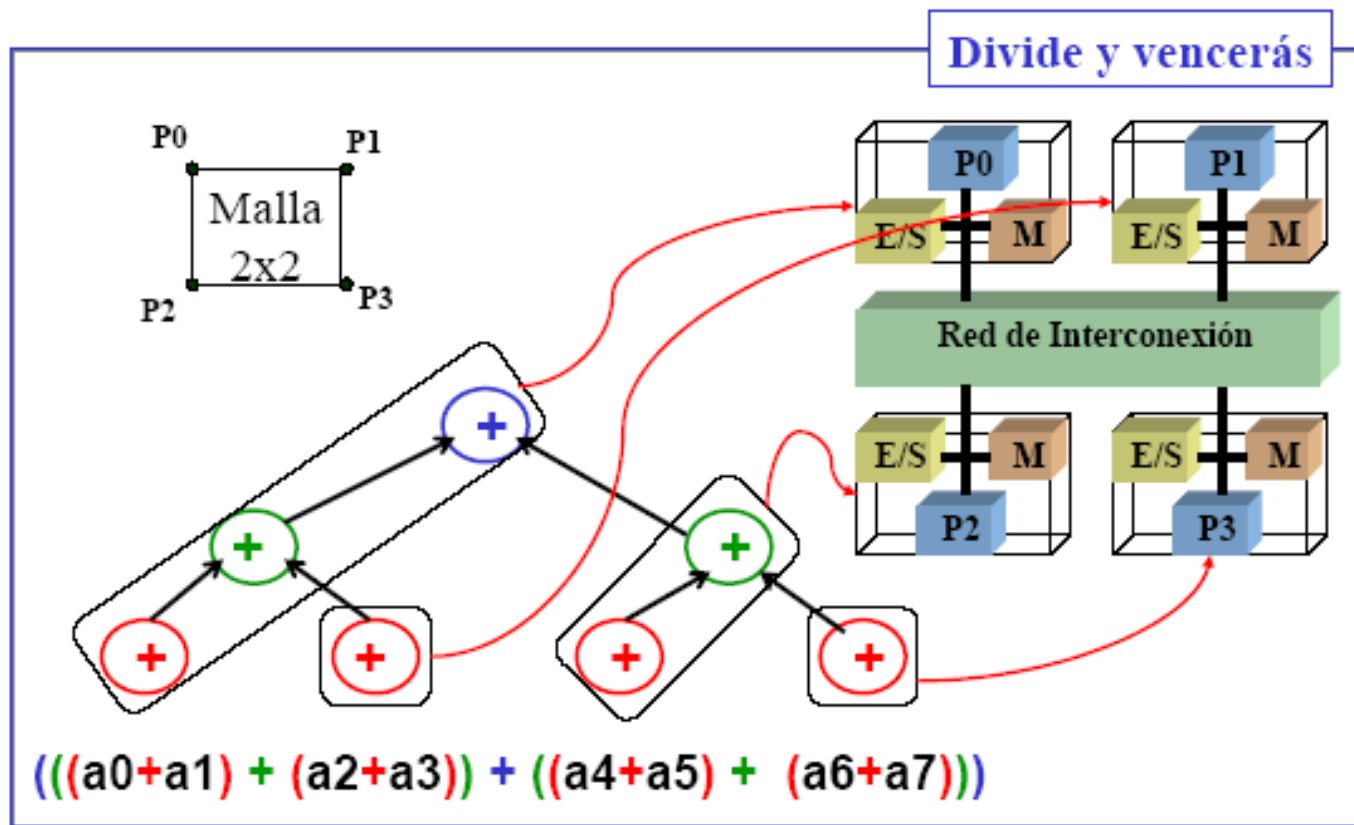
## Sesión 6. Paralelismo

Introducción

Conceptos

Prog. Paralela

- Estructuras de paralelismo: divide y vencerás (descomposición recursiva)



# Ingeniería de los Computadores

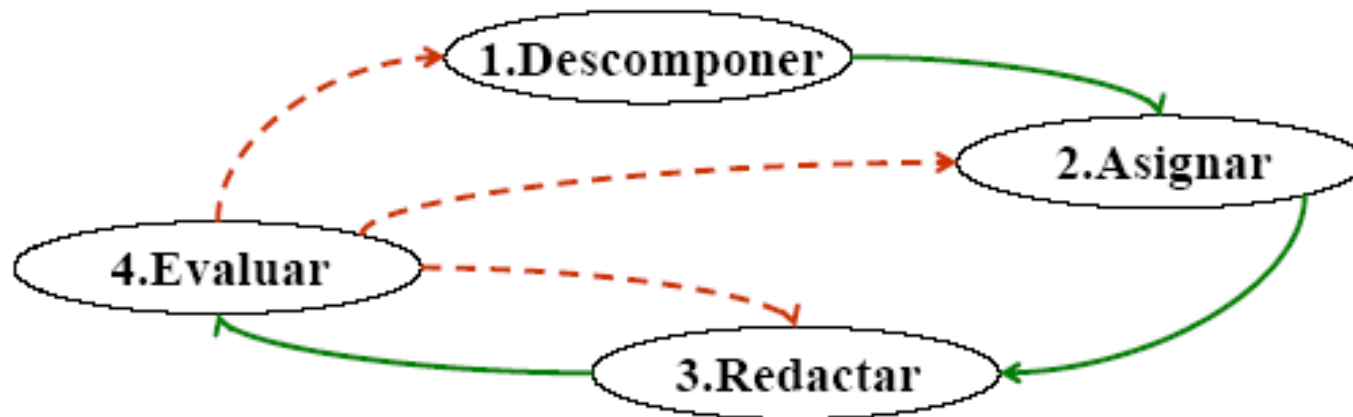
## Sesión 6. Paralelismo

Introducción

Conceptos

Prog. Paralela

- Proceso de paralelización
  - Descomposición en tareas independientes
  - Asignación de tareas a procesos y/o hebras
  - Redacción de código paralelo
  - Evaluación de prestaciones



# Ingeniería de los Computadores

## Sesión 6. Paralelismo

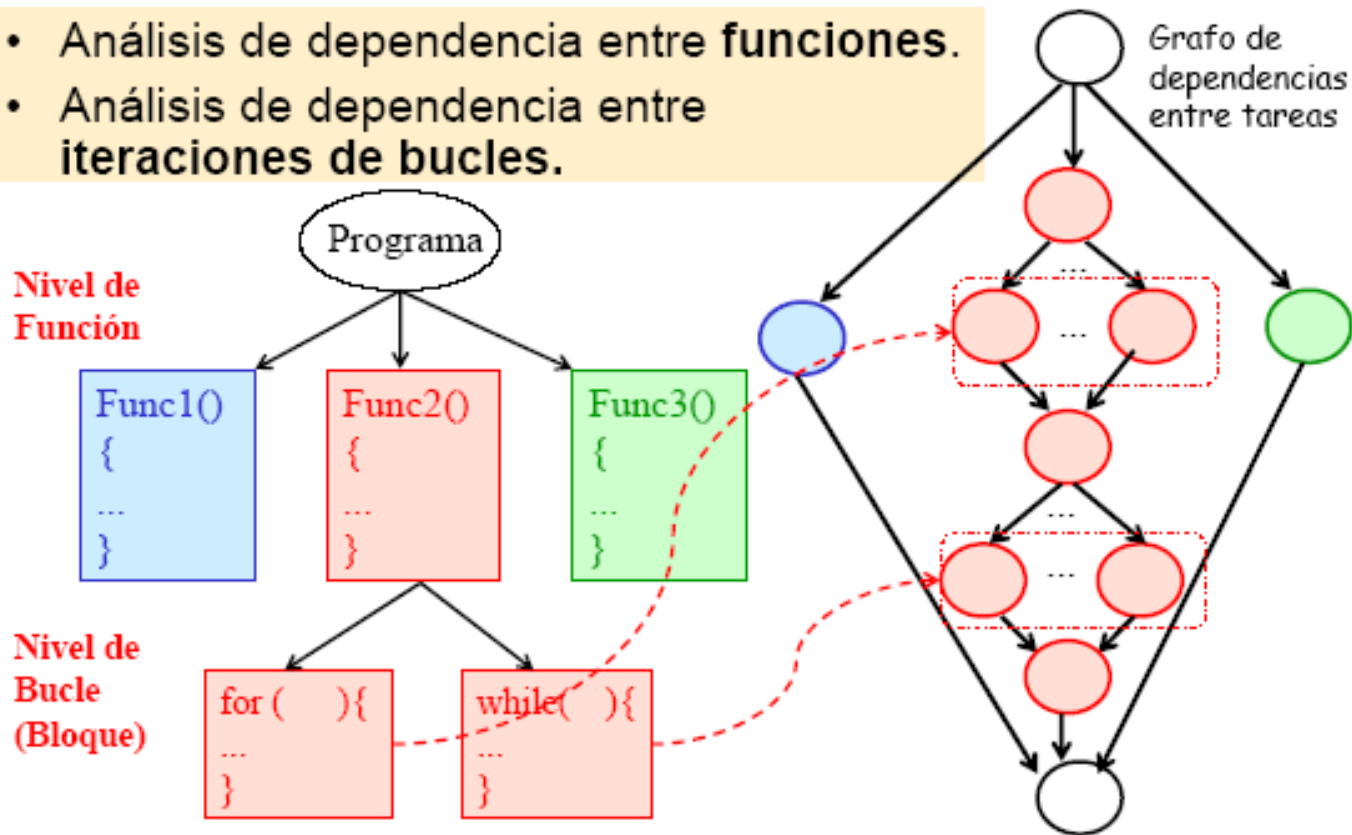
Introducción

Conceptos

Prog. Paralela

- Proceso de paralelización: descomposición en tareas independientes

- Análisis de dependencia entre **funciones**.
- Análisis de dependencia entre iteraciones de **bucles**.



# Ingeniería de los Computadores

## Sesión 6. Paralelismo

Introducción

Conceptos

Prog. Paralela

- Proceso de paralelización: asignación de tareas
  - Normalmente se asignan iteraciones de un ciclo a hebras y funciones a procesos
  - La granularidad depende de
    - El número de procesadores
    - El tiempo de comunicación/sincronización frente al tiempo de cálculo
  - Equilibrio en la carga de trabajo (que unos procesadores no esperen a otros)
  - Tipos de asignación
    - Dinámica (en tiempo de ejecución). Si no se conoce el número de tareas
    - Estática (programador o compilador)

# Ingeniería de los Computadores

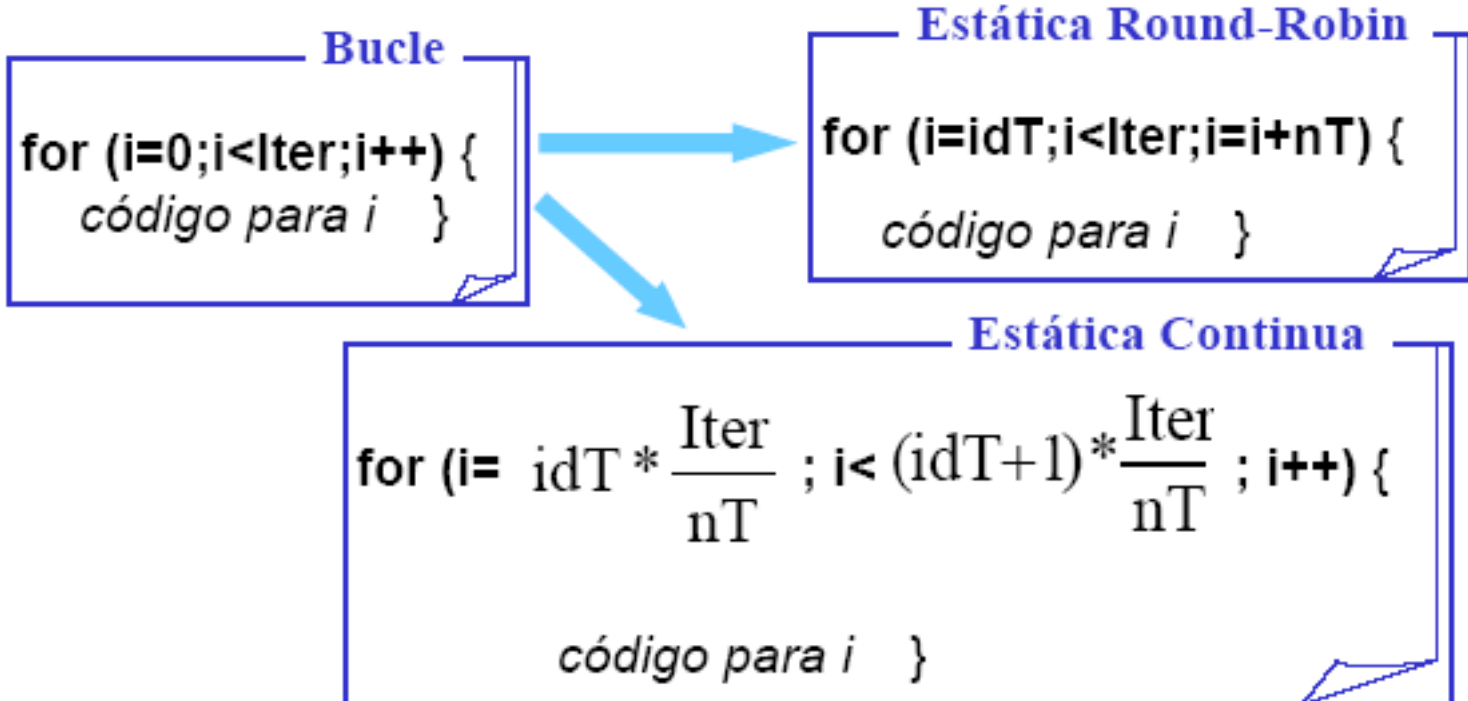
## Sesión 6. Paralelismo

Introducción

Conceptos

Prog. Paralela

- Proceso de paralelización: asignación de tareas
  - Asignación estática



# Ingeniería de los Computadores

## Sesión 6. Paralelismo

Introducción

Conceptos

Prog. Paralela

- Proceso de paralelización: asignación de tareas
  - Asignación dinámica

**Bucle**

```
for (i=0;i<lter;i++) {  
    código para i  
}
```



**Dinámica**

```
lock(k);  
    n=i; i=i+1;  
unlock(k);  
while (n<lter) {  
    código para n ;  
    lock(k);  
    n=i; i=i+1;  
    unlock(k);  
}
```



- Proceso de paralelización: redactar código paralelo
  - Depende de:
    - Estilo de programación: paso de mensajes, etc.
    - Modo de programación
    - Situación inicial
    - Herramienta utilizada para explicitar el paralelismo
    - Estructura del programa
  - Un programa paralelo debe incluir
    - Creación y destrucción de procesos/hebras
    - Asignación de carga de trabajo
    - Comunicación y sincronización

# Ingeniería de los Computadores

## Sesión 6. Paralelismo

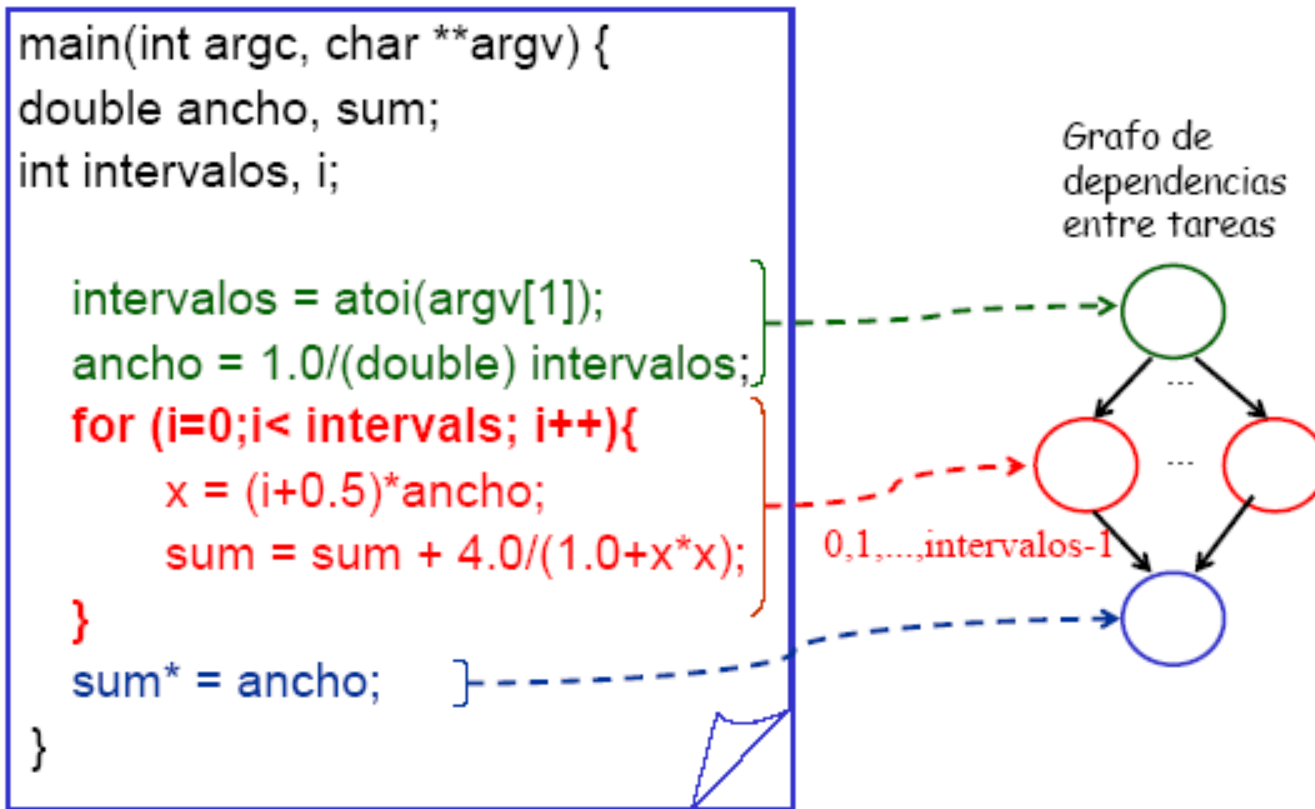
Introducción

Conceptos

Prog. Paralela

Ejemplo

- Cálculo de Pi por descomposición de tareas



# Ingeniería de los Computadores

## Sesión 6. Paralelismo

Introducción

Conceptos

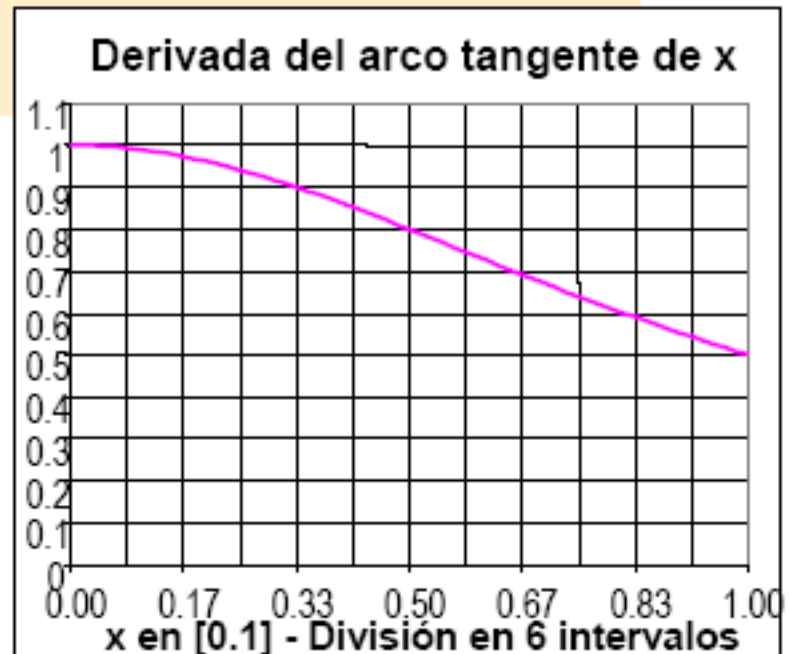
Prog. Paralela

Ejemplo

- Cálculo de Pi por descomposición de tareas

$$\left. \begin{array}{l} \operatorname{arctg}'(x) = \frac{1}{1+x^2} \\ \operatorname{arctg}(1) = \frac{\pi}{4} \\ \operatorname{arctg}(0) = 0 \end{array} \right\} \Rightarrow \int_0^1 \frac{1}{1+x^2} = \operatorname{arctg}(x) \Big|_0^1 = \frac{\pi}{4} - 0$$

- Pi se puede calcular por integración numérica.



- Cálculo de Pi por descomposición de tareas
  - Asignación estática de iteraciones del bucle (asignación Round Robin)
  - Redacción de código paralelo
    - Estilo de programación: paso de mensajes
    - Modo de programación: SPMD
    - Situación inicial: versión secuencial
    - Herramienta: MPI
    - Estructura del programa: paralelismo de datos o divide y vencerás

# Ingeniería de los Computadores

## Sesión 6. Paralelismo

Introducción

Conceptos

Prog. Paralela

Ejemplo

- Cálculo de Pi por descomposición de tareas

```
#include <mpi.h>
main(int argc, char **argv) {
    double ancho,x, sum, tsum; int intervalos, i; int nproc, iproc;

    if (MPI_Init(&argc, &argv) != MPI_SUCCESS) exit(1);
    MPI_Comm_size(MPI_COMM_WORLD, &nproc);
    MPI_Comm_rank(MPI_COMM_WORLD, &iproc);
    intervalos = atoi(argv[1]); ancho = 1.0 / (double) intervalos; lsum = 0;

    for (i=iproc; i<intervalos; i+=nproc) {
        x = (i + 0.5) * ancho; lsum += 4.0 / (1.0 + x * x);
    }
    lsum *= ancho;
    MPI_Reduce(&tsum, &sum, 1, MPI_DOUBLE,
               MPI_SUM,0,MPI_COMM_WORLD);
    MPI_Finalize();
}
```

**Enrolar**

**Asignar/Localizar**

**Comunicar/sincronizar**

**Desenrolar**

- Cálculo de Pi por descomposición de tareas
  - Asignación dinámica de iteraciones del bucle
  - Redacción de código paralelo
    - Estilo de programación: directivas
    - Modo de programación: SPMD
    - Situación inicial: versión secuencial
    - Herramienta: OpenMP
    - Estructura del programa: paralelismo de datos o divide y vencerás

# Ingeniería de los Computadores

## Sesión 6. Paralelismo

Introducción

Conceptos

Prog. Paralela

Ejemplo

- Cálculo de Pi por descomposición de tareas

```
#include <omp.h>
#define NUM_THREADS 4
main(int argc, char **argv) {
    double ancho,x, sum; int intervalos, i;
    intervalos = atoi(argv[1]); ancho = 1.0/(double) intervalos;

    omp_set_num_threads(NUM_THREADS);
    #pragma omp parallel

    #pragma omp for reduction(+:sum) private(x)
    schedule(dynamic)
    for (i=0;i< intervalos; i++) {
        x = (i+0.5)*ancho; sum = sum + 4.0/(1.0+x*x);
    }
    sum* = ancho;
}
```

**Crear/ Terminar**

**Comunicar/ sincronizar**

**Localizar**

**Asignar**