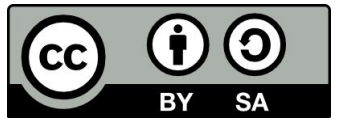
Computación de Altas Prestaciones

Tipos de aplicaciones paralelas

José Luis Risco Martín
Dpto. Arquitectura de Computadores y Automática
Universidad Complutense de Madrid



Índice

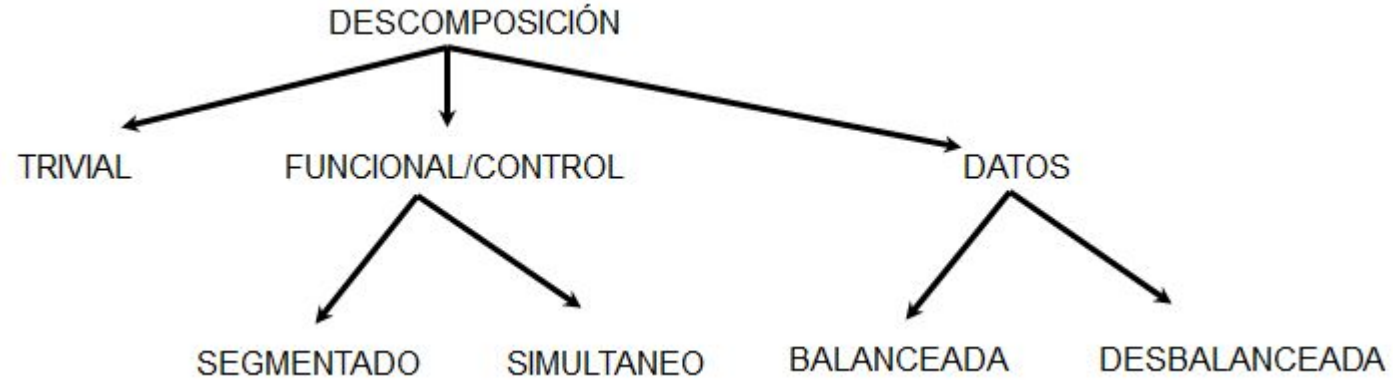
Introducción

Introducción (1/2)

- Hablamos de paralelización explícita ⇒ grano grueso
- La naturaleza del problema nos informa de la eficiencia de su implementación paralela:
  - Acceso a los datos (localidad)
  - Grado de concurrencia (paralelismo)
    - Cambio de algoritmo a otro más paralelo aunque sea menos rápido secuencialmente
- Por tanto, debemos analizar la aplicación para saber si su paralelización tendrá o no éxito
  - Debemos saber si el esfuerzo invertido en la paralelización merecerá la pena

Introducción (2/2)

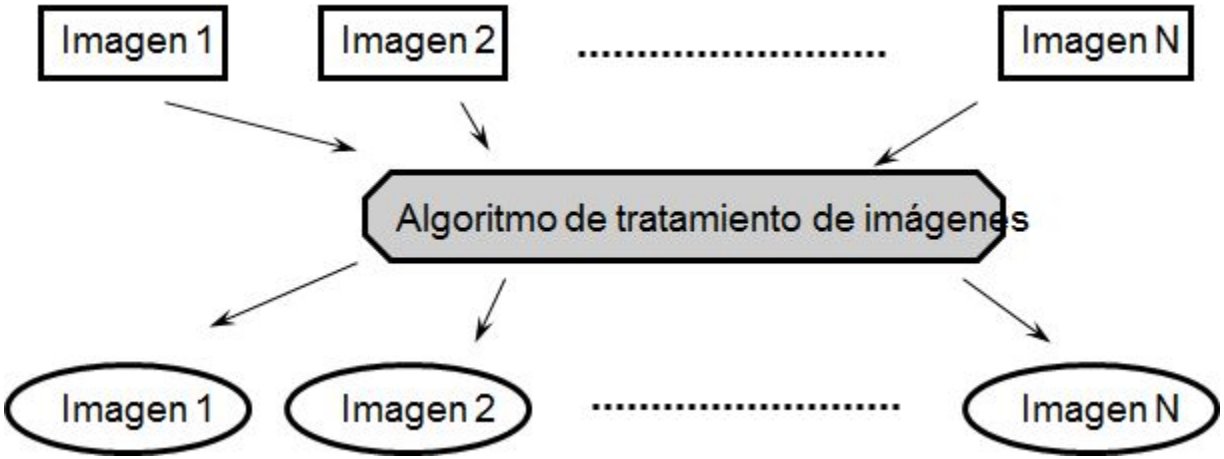
- Las tipos de aplicaciones paralelas es lo mismo que los tipos de descomposición de un problema sobre un conjunto de procesadores o paralelización (no tiene porque ser único)



- La paralelización de una aplicación se suele realizar combinando varios tipos
  - Paralelización en varios niveles: descomposición en dominios y de bucles

Paralelismo a nivel de tarea o trivial
Visión general (1/2)

- La aplicación consta de una serie de tareas que se pueden realizar simultáneamente y de manera independiente
- Características:
  - Cada tarea se podría realizar en un computador diferente
  - Cada tarea tiene su propio conjunto de datos de entrada
- Ejemplo:
  - Un algoritmo que actúa sobre varios conjuntos de datos de entrada.
  - Algoritmo de convolución sobre varias imágenes, resolutor de sistemas de ecuaciones con varias partes derechas, ...



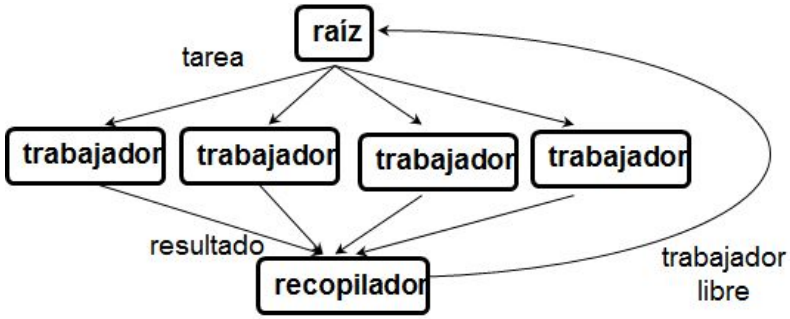
Visión general (2/2)

- Ventajas:
  - Se obtienen buenos rendimientos
    - No hay parte secuencial
    - No hay sobrecarga de comunicaciones
    - No hay desigualdad de carga
  - El tiempo invertido en la paralelización es mínimo
    - Se pueden usar algoritmos secuenciales
    - No hace falta usar un lenguaje de programación paralelo
- Inconvenientes:
  - Realmente no se puede considerar que el problema haya sido paralelizado sino replicado
  - Si el problema no es hacer muchas simulaciones sino una más rápida o precisa, se debe paralelizar

Granjas de procesos (1/2)

Sistema formado por los siguientes tres tipos de procesos:

- Raíz:
  - Encargado de dividir las tareas entre los trabajadores
  - Envía una nueva tarea a un trabajador cuando el recopilador le informa de que este ha terminado
- Trabajador:
  - Recibe la tarea del raíz
  - Procesa la tarea
  - Envía los resultados al recopilador
- Recopilador:
  - Recibe los resultados de los trabajadores
  - Informa al raíz de que un trabajador se ha quedado ocioso

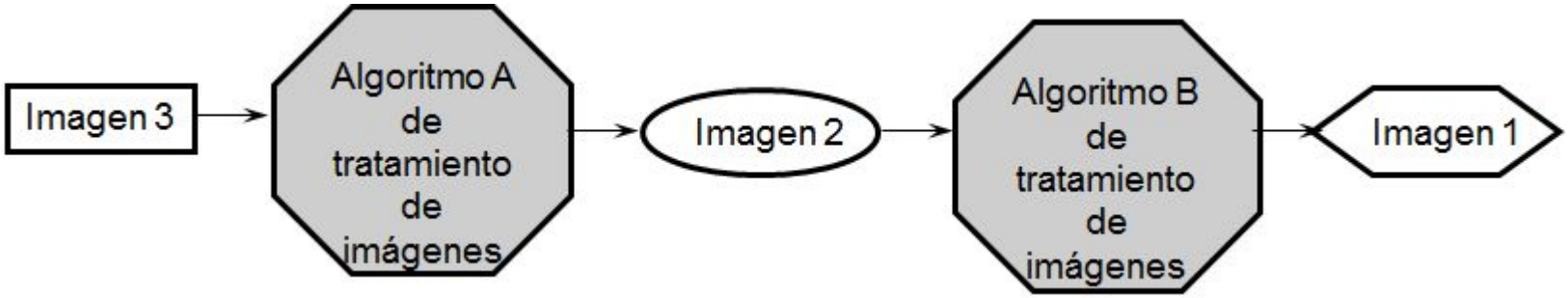


Granjas de procesos (2/2)

- Se suele emplear cuando se quieren realizar varias tareas independientes que requieren diferente cantidad de computación ⇒ Balanceo de carga dinámico
- Limitaciones y mejoras al modelo básico
  - Para evitar que los trabajadores se queden ociosos mientras esperan otra tarea, se coloca un buffer en el trabajador
    - Un trabajador mantiene dos tareas: sobre la que trabaja y la siguiente
    - Aumento de la productividad
  - Reparto de tareas para que todos los trabajadores terminen a la vez
    - Una posible solución es repartir primero las tareas más pesadas
  - Los trabajadores se comunican entre sí
    - Gestión muy complicada porque no se conoce a priori que trabajador desarrollará cada tarea

Paralelismo funcional/control
Segmentado (1/2)

- La aplicación realiza una única tarea que se puede segmentar en etapas
  - El paralelismo se introduce solapando el procesamiento de varios conjuntos de datos de entrada
  - Los resultados parciales se transmiten entre las etapas del cauce o pipe
- Características:
  - La aplicación se debe de poder dividir en etapas
- Ejemplo:
  - Fases de lectura y escritura de ficheros
  - Varias fases en el tratamiento de una imagen

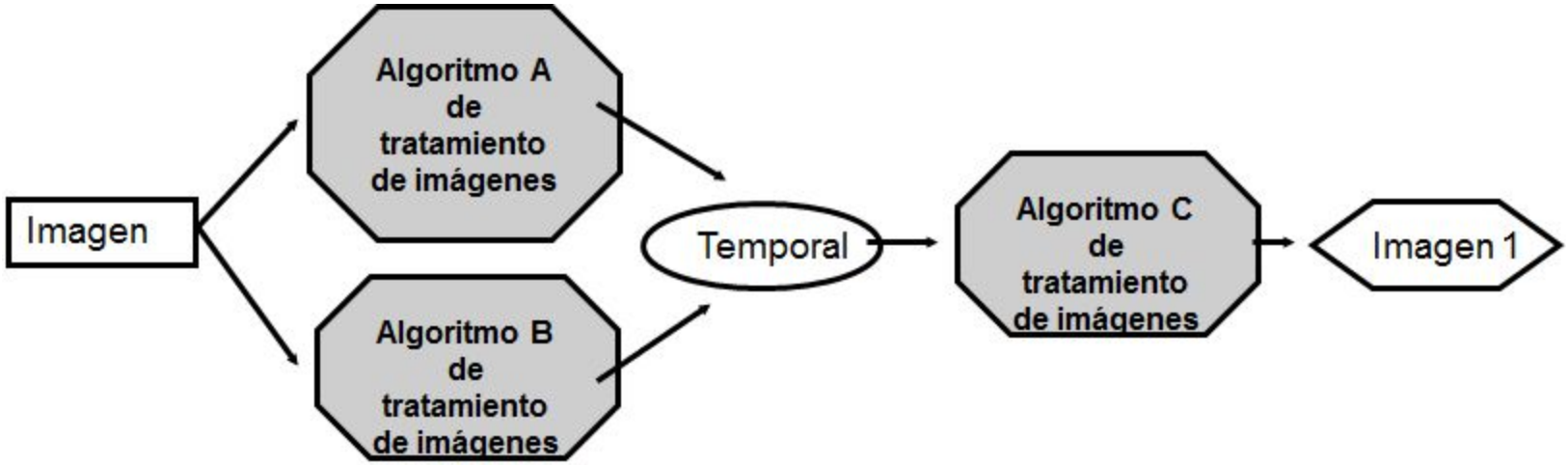


Segmentado (2/2)

- Ventajas:
  - Paralelización sencilla y natural
- Inconvenientes:
  - Las ineficiencias son debidas:
    - Al tiempo consumido en el llenado del cauce ⇒ tiempo de inicialización
    - A la desigualdad de la duración de las diferentes etapas
      - ✓ Posibles soluciones:
      - ✓ Usar CPUs más rápidas para las etapas más pesadas
      - ✓ Paralelizar la etapa más costosa (combinación de paralelismos)
  - Escalabilidad: El número de procesadores que se puede usar viene limitado por el número de etapas de la aplicación
    - Posible solución:
    - Combinar este paralelismo con otro asignando varios procesadores a cada etapa

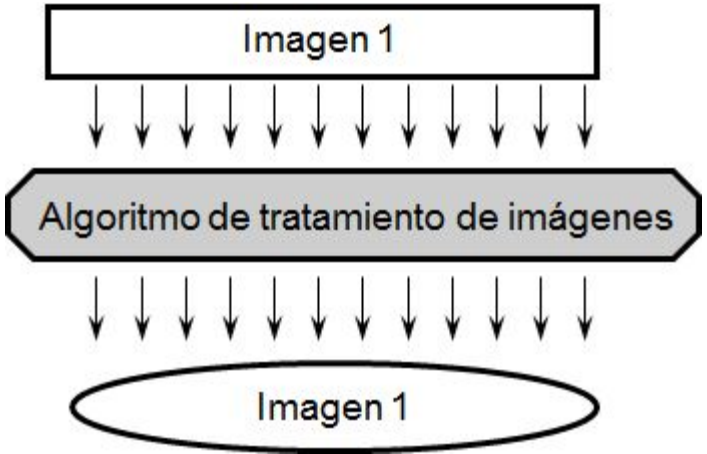
Simultáneo

- La aplicación realiza una única tarea que se compone de varias funciones o fragmentos de código que se pueden ejecutar en paralelo
- Características:
  - La aplicación se debe de poder dividir en funciones independientes
- Ejemplo:
  - Varias fases en el tratamiento de una imagen



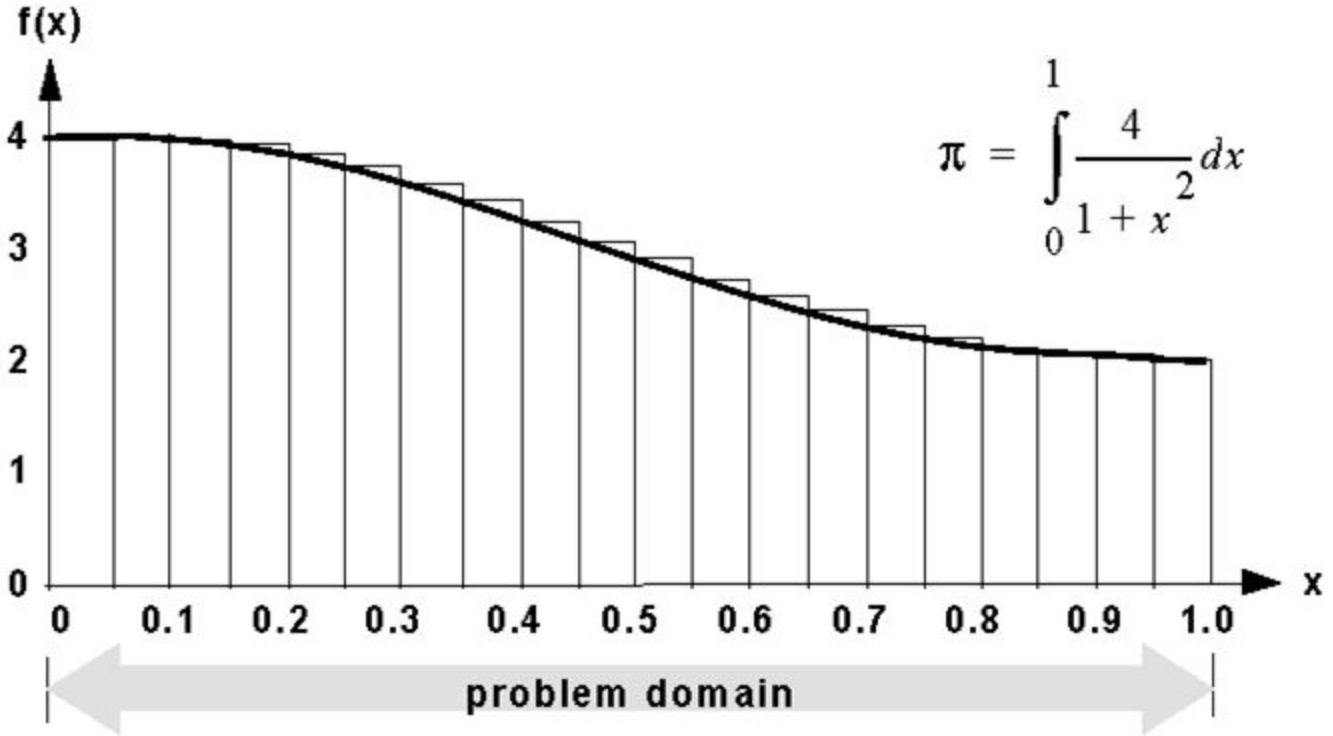
Paralelismo a nivel de datos
Totalmente síncrono o balanceado (1/2)

- Una operación se realiza de forma síncrona sobre todos los datos, los nuevos datos dependen solo de los antiguos
  - Los datos se reparten entre las procesadores uniformemente
- Las ineficiencias son debidas:
  - A que el número de datos no es múltiplo del número de procesadores
  - Tiempo consumido en la comunicación
- Requiere más trabajo de programación que los casos anteriores
- Ejemplo:
  - Un algoritmo de convolución sobre una única imagen



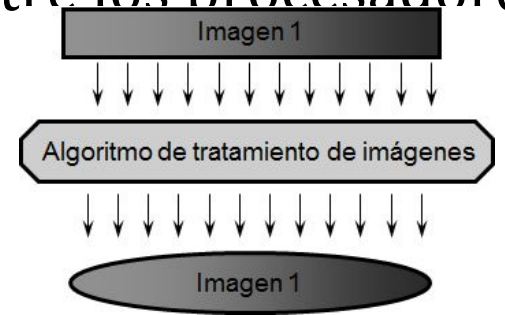
Totalmente síncrono o balanceado (2/2)

Ejemplo: Cálculo de una integral



Débilmente síncrono o desbalanceado (1/2)

- Cada procesador realiza una parte de un problema heterogéneo, se produce una sincronización y se vuelve a tratar el problema
  - Cuando termina un paso temporal algunos procesadores se mantienen en espera hasta que terminen todos, se intercambia información y se continúa con el siguiente paso temporal
- Las ineficiencias son debidas a:
  - La desigualdad dinámica del trabajo en cada procesador
  - Al tiempo consumido en la interacción entre los procesadores
- Es la más difícil de programar porque combina las dificultades de los paralelismos segmentado y totalmente síncrono (en este caso la interacción entre los procesadores no es tan simple)
- Ejemplos:
  - Dinámica molecular, astrofísica, ...



Débilmente síncrono o desbalanceado (2/2)

Ejemplo: Cálculo conjunto de Mandelbrot

```
for m = 0 ... 1023 do
 for n = 0 ... 1023 do
   c = x+yj
   a = 0+0j
   while ((k<1000) and (abs(a)<2)) do
     a = a^2+c
     k = k+1
   end
   data(n,m) = k
 end
end
```