Paralelismo, Arquitecturas Paralelas y Flynn

Parallel Architectures by Flynn

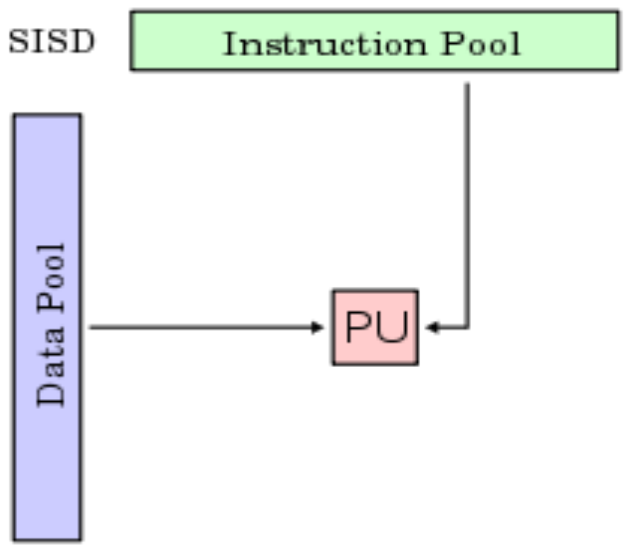
- *"...Parallel or concurrent operation has* ***many different forms*** *within a computer system..."*
- *"...A stream is a sequence of objects such as* ***data****, or of actions such as* ***instructions****.* ***Each stream is independent*** *of all other streams, and each element of a stream* ***can consist of one or more objects or actions****..."*

Las arquitecturas más comúnes según la cantidad de *streams* son:

- SISD.
- SIMD.
- MISD.
- MIMD.

SISD

- Significa *Single Instruction Single Data*.
- Arquitectura tradicional de un único procesador.
- Utiliza *pipelining*, realizando concurrentemente diferentes fases de procesamiento de una instrucción.
- Implementa *instruction level parallelism* (ILP) como superescalar o *very long instruction word* (VLIW).
- No se obtiene concurrencia de ejecución, pero si de procesos.



Paralelismo, Arquitecturas Paralelas y Flynn

SIMD

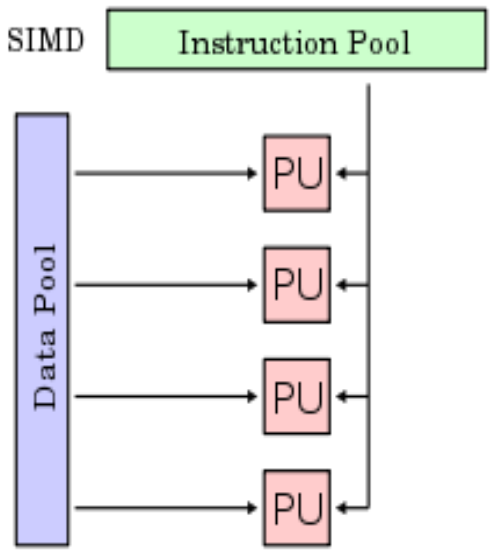
- Significa *Single Instruction Multiple Data*.
- Incluye procesadores de arreglo (*array*) y vectoriales.
  - Procesadores de arreglo: Instrucciones operan en múltiples elementos al mismo tiempo. Se conocen como *massively parallel processor*.
  - Procesadores vectoriales: Instrucciones operan en múltiples elementos en tiempos consecutivos.

LD VR ← A[3:0]
ADD VR ← VR,1
MUL VR ← VR,2
ST A[3:0] ← VR

SIMD (Tiempo vs Espacio)

LD0	LD1	LD2	LD3
AD0	AD1	AD2	AD3
MU0	MU1	MU2	MU3
ST0	ST1	ST2	ST3

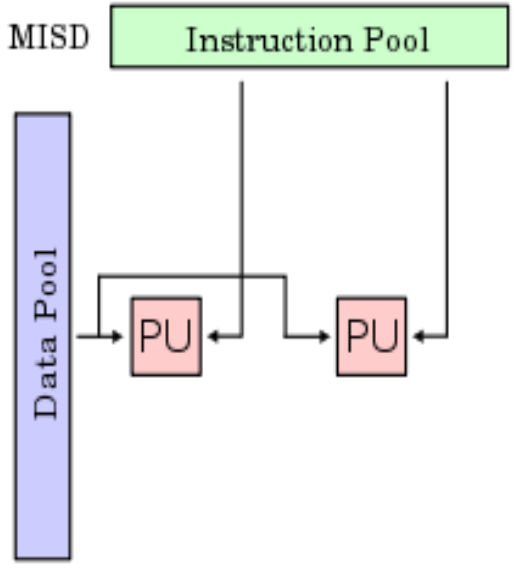
LD0			
LD1	AD0		
LD2	AD1	MU1	
LD3	AD2	MU2	ST0
	AD3	MU3	ST1
		MU4	ST2
			ST3



Paralelismo, Arquitecturas Paralelas y Flynn

MISD

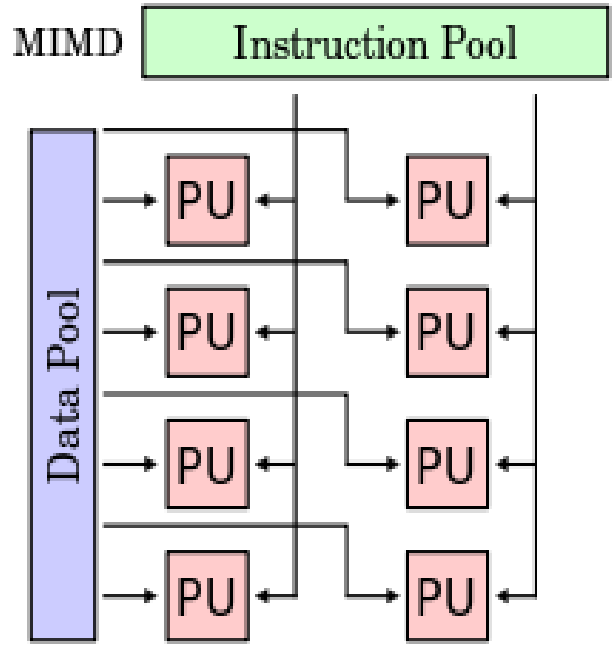
- Significa *Multiple Instruction, Single Data*.
- Se usa en sistemas aeroespaciales y arreglos sistólicos.
- También se puede usar para detectar y enmascarar errores



Paralelismo, Arquitecturas Paralelas y Flynn

MIMD

- Significa *Multiple Instruction, Multiple Data*.
- No necesariamente todos los procesadores deben ser idénticos, cada uno opera independientemente.
- Son: procesadores multinúcleo y superescalares.
- Cuando usan memoria compartida en este tipo de sistemas hay dos problemas:
  - Consistencia de memoria (se resuelve a través de combinación de técnicas de hardware y software).
  - Mantener la coherencia de cach´ (se resuelve mediante técnicas de hardware).



BreakoutRoom : Pregunta de examen

Paralelismo, Arquitecturas Paralelas y Flynn

Según Flynn cómo se catalogan:

$$[f(x), g(y), h(z)] = \left[\frac{x+1}{2}, \frac{\sin y}{y}, e^z\right] \quad (1)$$

$$[h(x, y)] = \left[e^{x+y}\right] \quad (2)$$

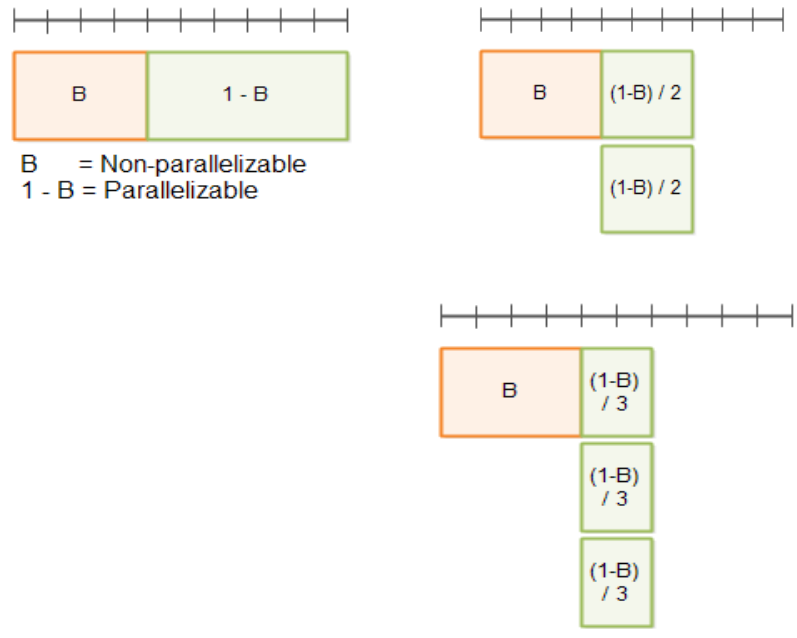
$$[f(x)] = a_o + x(a_1 + x(a_2 + a_3 x)) = a_0 + a_1 x^2 + a_2 x^3 \quad (3)$$

$$[g(x, y, z)] = x^{a_0} + y^{a_1} + z^{a_2} \quad (4)$$

Ley de Amdahl

¿Qué es?

- Permite obtener la ganancia en el desempeño debido a la mejora en una característica determinada.
- La ley de Amdahl puede servir como una guía para determinar la mejora real y como distribuir los recursos para tener mejor relación costo-desempeño.
- Gene Amdahl establece que todo programa se divide en:
  - Partes paralelizable.
  - Partes **no** paralelizables.



Ley de Amdahl

¿Qué es?

$$T(N) = B + \frac{(T-B)}{N}$$

Donde:

- B: es la parte no paralelizable.
- T: es el tiempo de duración de una tarea.

La fracción paralelizable está dada por un factor N

Speedup

$$\text{Speedup} = \frac{\text{Tiempo de ejecución de una tarea sin mejora}}{\text{Tiempo de ejecución de una tarea con mejora}}$$

$$\text{Speedup} = \frac{T(1)}{T(N)} = \frac{T(1)}{B + \frac{(T(1)-B)}{N}}$$

Con $T(1) = 1$ (sin mejora):

$$\text{Speedup} = \frac{1}{B + \frac{(1-B)}{N}}$$

Speedup Overall

$$\text{Speedup}_{\text{overall}} = \frac{\text{Execution time}_{\text{old}}}{\text{Execution time}_{\text{new}}} = \frac{1}{(1-\text{Fraction}_{\text{enhanced}}) + \frac{\text{Fraction}_{\text{enhanced}}}{\text{Speedup}_{\text{enhanced}}}}$$

Donde

- $\text{Fraction}_{\text{enhanced}}$ es la fracción del tiempo de computaci original que se puede ver beneficiado por la mejora.
- $\text{Speed}_{\text{enhanced}}$ es la ganancia del producto de la ejecuci en modo "mejorado". Esto es, qué tan rápido ejecutaría la tarea si la mejora se aplicara a todo el programa.

Ejemplo: Ley de Amdahl

Supongamos que se desea mejorar un procesador utilizado para un servidor Web. El nuevo procesador es **10 veces más rápido** en tiempo de computación para la aplicación de servidor Web que el procesador antiguo. Asumiendo que el procesador original está **ocupado** un **40 %** del tiempo y el **60 %** del tiempo **esperando** por dispositivos de Entrada/Salida. ¿Cuál es la ganancia total producto de incorporar la mejora?

$$\text{Speedup}_{\text{overall}} = \frac{\text{Execution time}_{\text{old}}}{\text{Execution time}_{\text{new}}} = \frac{1}{(1-\text{Fraction}_{\text{enhanced}}) + \frac{\text{Fraction}_{\text{enhanced}}}{\text{Speedup}_{\text{enhanced}}}}$$

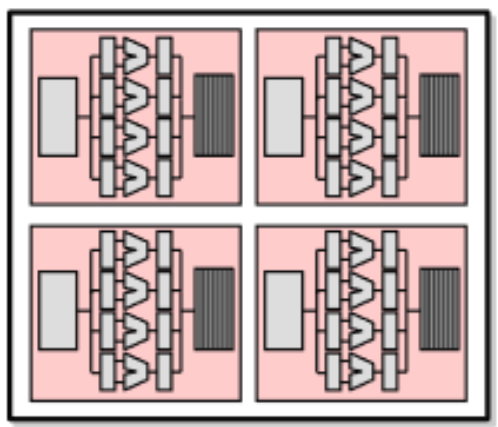
$$\text{Fraction}_{\text{enhanced}} = 40\,\% = 0{,}4 \quad \text{Speedup}_{\text{enhanced}} = 10$$

$$\text{Speedup}_{\text{overall}} = 1{,}56$$

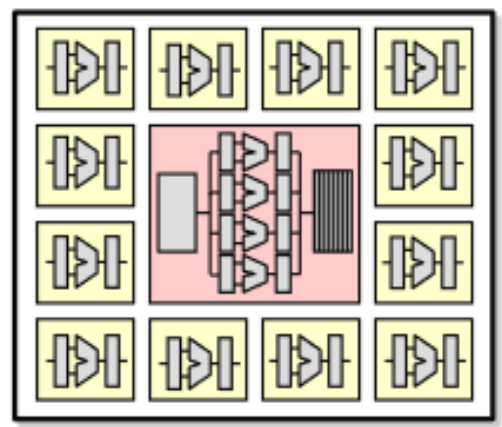
Ley de Amdahl

Amdahl con unidades de procesamiento simétricas y no-simétricas

- La paralelización es uniformemente distribuida en las unidades de ejecución (procesadores).
- No aplica en sistemas heterogéneos y *multicore* (sim étricos y no simétricos).
- Supone que no hay conflictos de recursos.



Symmetric: Four 4-BCE cores



Asymmetric: One 4-BCE core & Twelve 1-BCE base cores

Confiabilidad

Probabilidad de que el sistema esté funcionando en el instante t dado que funcionaba en el instante t_0. Se tiene:

- Tiempo medio para una falla (MTTF).
- Fallos por tiempo (λ): $\lambda = \frac{1}{\text{MTFF}}$

R(t): probabilidad de que un sistema falle en t unidades de tiempo después de la última falla.

$$R(t) = e^{-\lambda(t-t_0)}$$

Mantenibilidad

Tiempo requerido para que el sistema este funcionando luego que se dio una falla.

- Tiempo medio para reparar (MTTR): Tiempo de la interrupción del servicio.
- Tasa de reparación (μ): $\mu = \frac{1}{\text{MTTR}}$

M(t): probabilidad de que un sistema este funcionando en t unidades de tiempo después de que se presentó una falla.

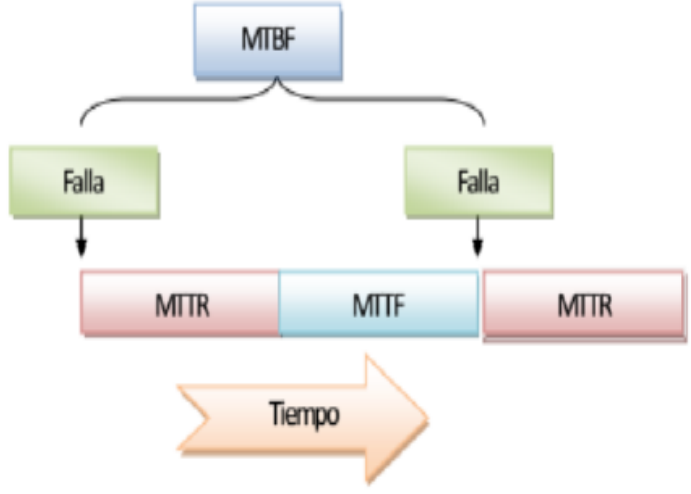
$$M(t) = e^{-\mu t}$$

Disponibilidad

Es el porcentaje del tiempo en el que el sistema estará disponible para brindar un servicio correcto.

Tiempo medio entre fallas (MTBF): MTTF + MTTR

$$A = \frac{\text{MTTF}}{\text{MTBF}}$$

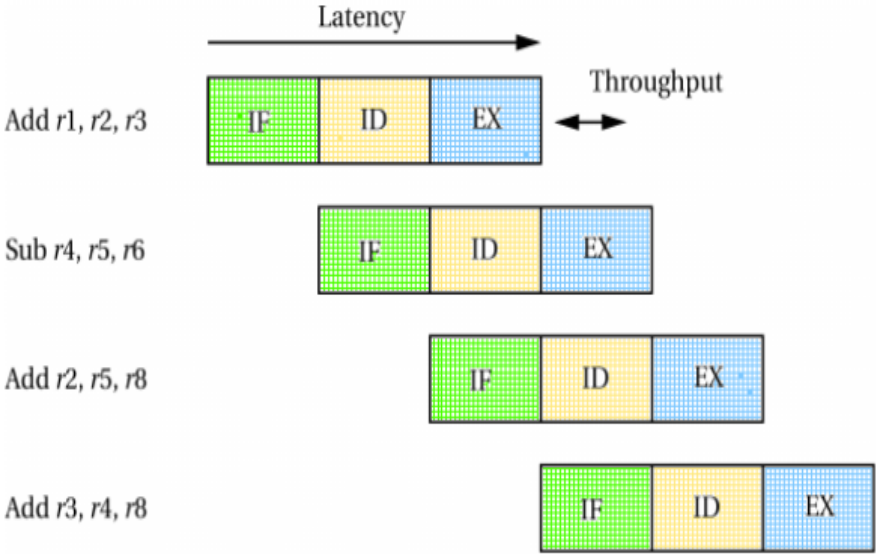


Benchmarking

El benchmark es un instrumento para comparar el desempeño de varios sistemas en aplicaciones reales.

Representa un recurso de software para evaluar un sistema y discriminar la mejor opción para el diseño.

Varios tipos de benchmarks: SPEC, EEMBC, BDTi, Drystone, CoreMark.



Término aplica de manera diferente según el campo:

- ISP: Calidad de imagen.
- Memorias: Accesos a memoria por segundo.
- CPU's: Medida de la tasa en que los programas son ejecutados (IPC, CPI).

Punto de vista microscópico:

- Latencia: Tiempo requerido para ejecutar una instrucción desde inicio hasta finalización.
- Flujo de instrucciones (throughput): Tasa de finalización de instrucciones.