**Taller Taxonomía de Flynn y Eficiencia en**

**Computación Paralela**

Ejercicio 1. Taxonomía de Flynn – Clasificación

Clasifique los siguientes escenarios en SISD, SIMD, MISD o MIMD. Justifique cada

respuesta:

1. Una CPU Intel 8086 ejecutando un programa en secuencia.

**SISD**

1. Una GPU NVIDIA procesando miles de pixeles en paralelo.

**SIMD**

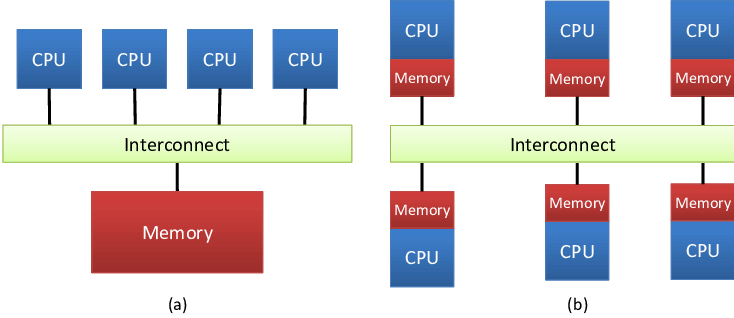
1. Un sistema de control en una nave espacial donde varios algoritmos distintos validan la misma señal de entrada.

**MISD**d) Un cluster de supercomputación ejecutando simulaciones meteorológicas distribuidas en varios nodos.

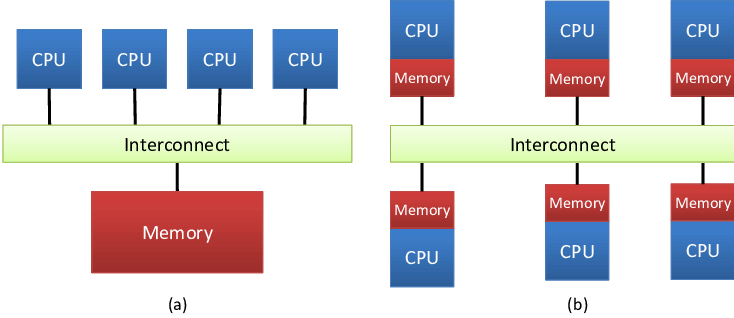
**MIMD**

Ejercicio 2. Taxonomía de Flynn – Variantes

Explique la diferencia entre:



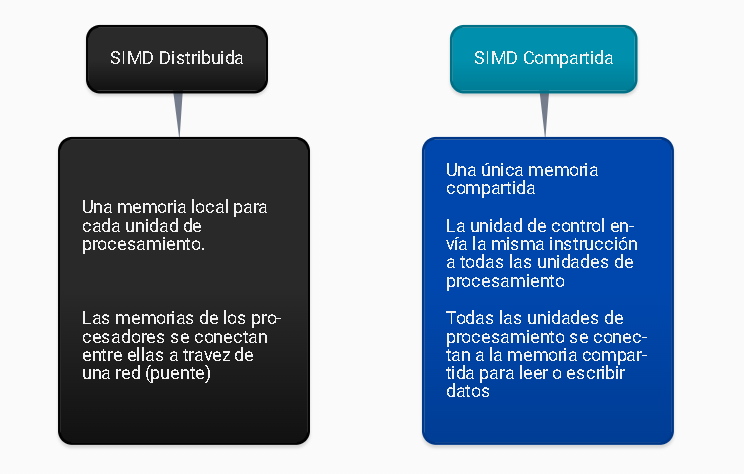
Mientras que la memoria compartida tiene conexión entre estos nodos sin tener latencia al tener la misma latencia, permitiendo por ejemplo el hacer una misma cosa a un dato



La memoria distribuida permite accesos únicos en los cuales el CPU no puede entrar a usar datos de cualquier otro CPU o nodo del mismo sistema

1. SIMD con memoria compartida vs SIMD con memoria distribuida.

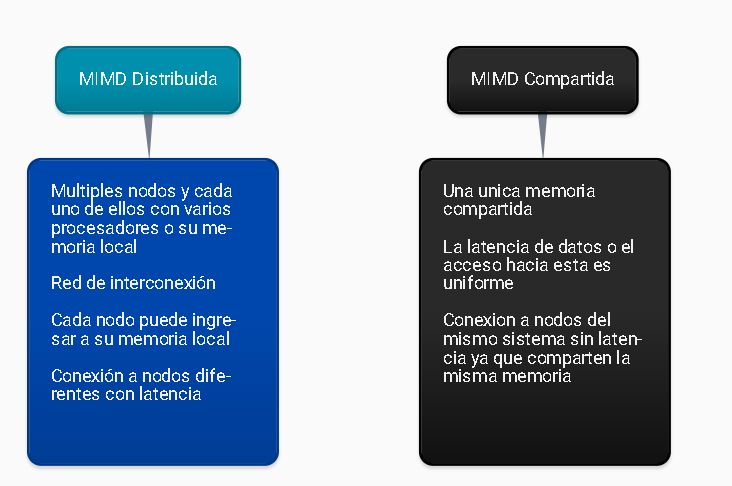
Esta taxonomía con memoria compartida es más común en GPU moderas en donde los núcleos se encargar de trabajar en el mismo espacio mientras que la memoria distribuida usa arquitecturas vectoriales



b) MIMD con memoria compartida (UMA) vs MIMD con memoria distribuida

(NUMA).

Entre estas taxonomías la MIMD con memoria distribuida esta enfocada hacia los super computadores mientras que las de memoria compartida son para servidores de multiprocesamiento de dato



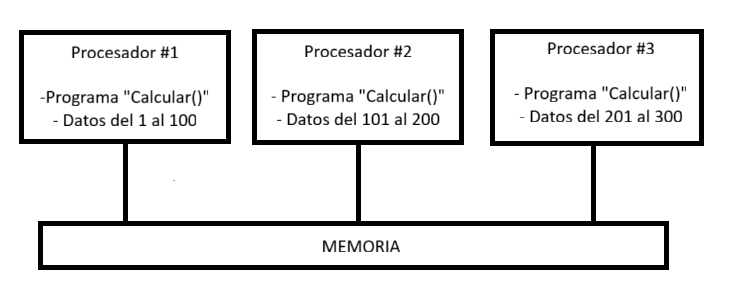
Ejercicio 3. Extensiones – SPMD y MPMD

Explique, con sus propias palabras y usando ejemplos sencillos:

1. ¿Qué significa que varios procesadores ejecuten el mismo programa, pero con datos diferentes (modelo SPMD)?

Los programas únicos con diferente información nos permiten hacer simulaciones de diferentes casos ya que, aunque utilicen el mismo programa de manera cíclica son capaces de manejar diferente información y dar resultados diferentes o en casos mas familiarizados con nuestra carrera la programación secuencial como la suma de dos matrices para retornar una única, es una instrucción repetida con diferente manejo de datos

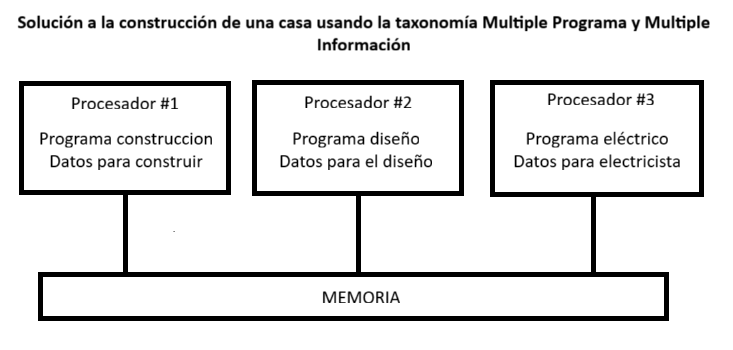
Ejemplo: varios estudiantes con la misma guía de ejercicios, pero cada uno resuelve un conjunto distinto de problemas.



1. ¿Qué significa que distintos procesadores ejecuten programas diferentes con datos diferentes (modelo MPMD)?

Esto es la unificación de varios programas que pueden hacer el funcionamiento correcto de uno solo o la armonía entre estos para que funcione uno principal, todo esto manejando diferentes tipos de datos, algo así puede ser una aplicación que muestre el modelo climático ya que utiliza una aplicación para calcular el modelo atmosférico, el modelo oceánico y el modelo de superficie y todos estos modelos cada uno utiliza por aparte datos diferente que son manipulados para al final conformar uno principal, también es usado para supercomputación

Ejemplo: un equipo de trabajo en el que un estudiante hace cálculos, otro genera gráficos y otro redacta el informe.



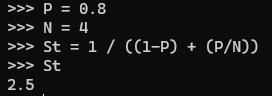
Ejercicio 4. Speedup teórico – Ley de Amdahl

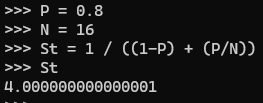
Suponga que el 80% de un programa puede paralelizarse, mientras que el 20% debe ejecutarse secuencialmente.



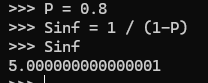
P = 0.8

1. Calcule el speedup teórico máximo usando 4 procesadores.



1. Calcule el speedup teórico máximo usando 16 procesadores.  
     
   

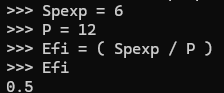
c) ¿Cual es el límite de speedup si los procesadores tienden a infinito?



Ejercicio 5. Eficiencia de un sistema paralelo

Un programa paralelo logra un speedup experimental de 6 al ejecutarse con 12 procesadores.

a) Calcule la eficiencia del sistema.  
  

b) ¿Qué significa ese valor en términos de aprovechamiento de recursos?

El valor de eficiencia es 0.5 y según los criterios de eficiencia cuando el valor es menor a 1 como lo es en este caso, hay perdida fuerte de rendimiento por lo cual es posible mejorar su paralización para obtener mejores resultados

c) ¿Recomendaría agregar más procesadores? Justifique.

Para este caso el agregar más procesadores solo hará que el valor de eficiencia se acerque mas a 0 por lo cual seria un peor rendimiento, en este caso es mejor que se haga una mayor paralización para que la diferencia de rendimiento en el valor experimental pueda llegar a ser mejor y así mismo su eficiencia

Ejercicio 6. Análisis práctico – OpenMP

Ejecute el siguiente código en C con OpenMP, el cual calcula numeros primos en

paralelo:

Archivo: primes number parallel.c

Compilación:

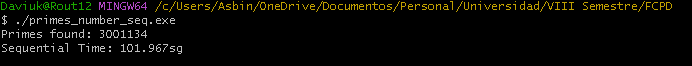
gcc primes\_number\_parallel.c -o primes\_number\_parallel -fopenmp

Ejecución:

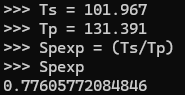
OMP\_NUM\_THREADS=12 ./primes\_number\_parallel.exe

Mida los tiempos de ejecución con diferente número de hilos (2, 4, 8, 12) y calcule:

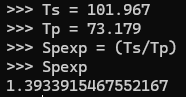
a) El speedup experimental para cada caso.  
Secuencial

  
2 hilos

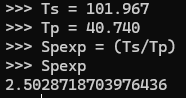


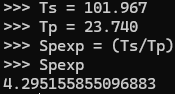
  
4 hilos



  
8 hilos

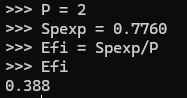


  
12 hilos

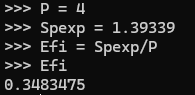
  


b) La eficiencia para cada caso.

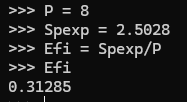
2 hilos



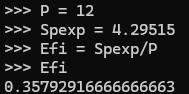
4 hilos



8 hilos



12 hilos

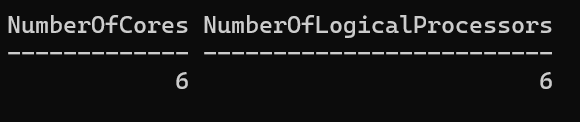


c) Compare los resultados con el speedup teórico esperado.

Todos los valores son valores pequeños lo cual significa que su eficiencia es menor a 1 y no esta dando buen rendimiento, esto puede ser por el computador o por temas de la programación del propio código, los resultados teóricos vienen a dar valores mayores mientras que experimentalmente son valores mucho menores

Ejercicio 7. Análisis Experimental – Primos Secuencial vs Paralelo

Ejecute nuevamente el programa primes number parallel.c para calcular números primos bajo las siguientes condiciones:



1. Caso 1: N = 50 millones de números.

Secuencial



Paralelo



1. Caso 2: N = 35 millones de números

Secuencial



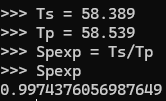
Paralelo



1. Para cada caso, ejecute el programa en versión secuencial y paralelo, utilizando todos los núcleos disponibles en su máquina.

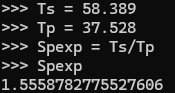
Para 35 millones de datos

1 hilo (secuencial)

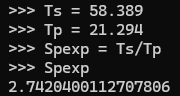




2 hilos

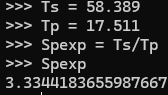




4 hilos



6 hilos (hilos máximos de la maquina)



Una vez se han realizado los datos y ejecutados en el libro de Jupyter analice y

responda.

1. El comportamiento del speedup experimental vs. speedup ideal.

Los resultados experimentales están muy por debajo de los esperados o de los ideales, esto se puede deber al rendimiento general del computador o a la ineficiencia de manera correcta de la programación del computador

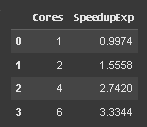
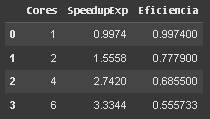
1. La eficiencia en función del número de núcleos.

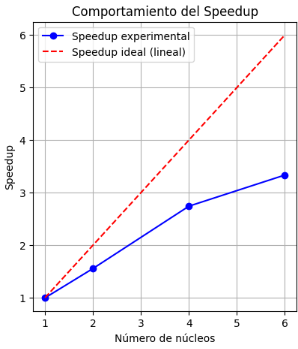
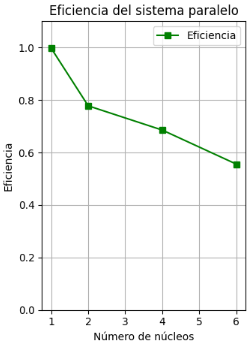
Es decreciente lo cual quiere decir que en este caso el código puede llegar a funcionar mejor de manera secuencial antes que funcionar de manera paralela

c) Interprete los resultados: ¿existe un punto de saturación? ¿Qué tan bien se

aprovechan los procesadores?

Desde los 4 núcleos el funcionamiento empieza a volverse recto sin embargo desde el inicio no funciona de la mejor manera la ejecución del código paralelizado para este caso





<https://colab.research.google.com/drive/14x2giB0JwS_v96vt3fiZPyssDk-t41h2?usp=sharing>