1. Paralelismo Implícito en Aplicaciones

- **Niveles de Paralelismo:**
  - **Programa:** Ejecución en paralelo de programas/procesos sin dependencias.
  - **Función:** Paralelización de funciones independientes.
  - **Bucle (Bloques):** Paralelización de iteraciones (hay que eliminar dependencias entre ellas).
  - **Operación:** Paralelismo a nivel de instrucciones; uso de SIMD para vectores.
- **Dependencias de Datos:**
  - **RAW (Read After Write):** Se necesita el resultado de una operación previa.
  - **WAW (Write After Write):** Dos escrituras en la misma dirección.
  - **WAR (Write After Read):** Se modifica un dato después de haberlo leído.
  - **Condición básica:** Mismo dato y orden: el bloque B1 debe ejecutarse antes que B2.
- **Granularidad:**
  - **Grano grueso:** Paralelismo entre programas completos.
  - **Grano medio:** Paralelismo entre bloques o funciones.
  - **Grano fino:** Paralelismo en bucles o incluso a nivel de instrucciones.
- **Paralelismo de Tareas vs. Datos:**
  - **Tareas:** Extraído de la estructura lógica (funciones); se paralelizan tareas enteras.
  - **Datos:** Operaciones sobre estructuras (vectores, matrices); cada operación escalar es independiente.

---

2. Clasificación de Arquitecturas Paralelas

- **Computación Paralela:**
  - Se realiza en un único sistema (multicore, SMP); se ve como una unidad autónoma.
- **Computación Distribuida:**
  - Utiliza recursos de múltiples nodos conectados en red (clusters, grid, cloud).
- **Clasificación Comercial:**
  - **Empotrados:** Aplicaciones específicas, restricciones en consumo, tamaño y tiempo real.
  - **PC/WS:** Uso general.
  - **Servidores:** Gama alta, media o baja.
  - **Supercomputadores:** Para tareas de alta demanda.
- **Clasificación de Flynn (1972):**
  - **SISD:** Un único flujo de instrucciones y de datos.
  - **SIMD:** Una instrucción, múltiples datos (ideal para vectorial).
  - **MISD:** Múltiples instrucciones, un único flujo de datos (rara vez se usa).
  - **MIMD:** Múltiples instrucciones y múltiples flujos de datos (la más general).

- **Sistemas de Memoria:**
  - **SMP:** Memoria centralizada; comunicación implícita (variables compartidas).
  - **Cluster:** Cada nodo tiene memoria local; comunicación explícita (por mensajes).
  - **NUMA:** Memoria físicamente distribuida, pero se programa como compartida.
  - **NORMA:** Acceso a memoria local y remota mediante instrucciones de carga/almacenamiento, pero con modelos diferenciados.
- **Arquitecturas según Paralelismo:**
  - **DLP (Data Level Parallelism):** Uso de unidades SIMD (operaciones vectoriales).
  - **ILP (Instruction Level Parallelism):** Procesadores segmentados, superescalares o VLIW.
  - **TLP (Thread Level Parallelism):** Paralelismo en hebras o procesos; se ve en multiprocesadores o clusters.

---

3. Evaluación de Prestaciones

- **Tiempo de Respuesta (Wall-clock Time):**
  - Tiempo total desde el inicio hasta el fin de la ejecución.
- **Tiempo de CPU:**
  - Tiempo en que el procesador ejecuta instrucciones (usuario + sistema).
- **Fórmula del Tiempo de CPU:**
  [T_{CPU} = NI \times CPI \times T_{ciclo}]
  - **NI:** Número de instrucciones.
  - **CPI:** Ciclos por instrucción.
  - **T(_{ciclo}):** Tiempo por ciclo = (1/) (frecuencia de reloj).
- **Unidades de Medida:**
  - **MIPS:** Millones de instrucciones por segundo.
  - **GFLOPS:** (\frac{\text{Operaciones de punto flotante}}{T \times 10^9}).
  - **GIPS:** Miles de millones de instrucciones por segundo.
- **Ejemplo de cálculo GFLOPS:**
  Si en un bucle se realizan (10^{12}) operaciones en 2 s,
  (GFLOPS = \frac{10^{12}}{2 \times 10^9} = 500) GFLOPS.

---

4. Ley de Amdahl

- **Fórmula:**
  [S \le \frac{1}{f + \frac{1-f}{p}}]
  - **p:** Factor de mejora (si se duplica, p = 2).
  - **f:** Fracción del tiempo que NO se beneficia de la mejora (0 ≤ f ≤ 1).
- **Clave:**
  - La mejora total nunca es igual a (p) si (f > 0).

5. Preguntas Tipo Test y Frases Clave

1. **Comunicación entre hebras:**
   - *Las hebras comparten memoria → no requieren llamadas al SO para comunicarse.*
2. **Ley de Amdahl:**
   - *La ganancia total nunca es igual a p si parte del tiempo (f) no se mejora.*
3. **Cálculo de GFLOPS (Ejemplo 1):**
   - Bucle: `a[i] = b[i]*c + a[i]`, N = (10^{14}), tiempo = 10 s.
   - 2 FLOP por iteración → Total = (2 \times 10^{14}) FLOP.
   - GFLOPS = (2\times10^{14} / (10 \times 10^9) = 20000) GFLOPS.
4. **Computador NUMA:**
   - *Memoria distribuida físicamente, pero se programa como compartida.* (VERDADERO)
5. **Computador NORMA:**
   - *Los accesos a memoria local y remota no se realizan de la misma forma.* (FALSO)
6. **Procesador superescalar:**
   - *Puede tener CPI < 1, ya que ejecuta varias instrucciones por ciclo.* (VERDADERO)
7. **Dependencia RAW/WAR en instrucciones:**
   - *RAW: i2 necesita el dato de i1; WAW: dos escrituras en el mismo registro; WAR: escribir tras leer.*
   - Ejemplo:
     - Secuencia (i1) add r1, r2, r3; (i2) sub r1, r1, r4 → Dependencia RAW (i2 usa r1 de i1).
     - En secuencias donde se usan distintos registros, la dependencia se reduce.
8. **Tiempo de ejecución (Programa de 2000M instrucciones):**
   - Si CPI promedio = 4 y reloj de 1 GHz →
     Tiempo = (2000 \times 10^6 \times 4 / 10^9 = 8) segundos.
9. **GIPS en núcleo superescalar:**
   - 4 instrucciones/ciclo a 2 GHz →
     (4 \times 2 = 8) GIPS.
10. **Otras preguntas de Test sobre dependencias:**
    - *"No hay dependencia WAR entre: (i1) add r1, r2, r3; (i2) sub r1, r1, r4" → VERDADERO.*
    - *"En secuencia: (i1) add r1, r2, r3; (i2) sub r1, r2, r4; (i3) add r3, r2, r1, no hay dependencia por r2" → VERDADERO.*
    - *"En secuencia: (i1) add r1, r2, r4; (i2) add r4, r2, r3; (i3) sub r1, r1, r4 → RAW por r4 entre i2 e i3" → VERDADERO.*

11. **Cálculo de tiempo (Programa de 1000M instrucciones, 5 tipos, 20% cada uno, 2 GHz):**
    - CPI promedio = (6+4+3+5+2)/5 = 4.
    - Total de ciclos = (10^9 \times 4 = 4 \times 10^9).
    - Tiempo = (4 \times 10^9 / (2 \times 10^9) = 2) segundos.
12. **Cluster vs. NUMA:**
    - *Cluster: comunicación explícita por mensajes.*
      NUMA: memoria distribuida pero modelo compartido.
    - Por ello, "Un cluster es un computador NUMA" → FALSO.
13. **SLURM – Opciones en srun/sbatch:**
    - *-n1 → crea un único proceso.*
    - *--cpus-per-task=X → asigna X núcleos por tarea.*
    - *--hint=nomultithread → evita usar hyperthreading (1 hebra por núcleo).*
    - Ejemplo:
      - Para 12 hebras en núcleos físicos distintos:
        `srun -p ac --account=ac -n1 --cpus-per-task=12 --hint=nomultithread HelloOMP`
      - Con `sbatch -p ac --account=ac -n1 --cpus-per-task=6 script.sh` se usan 6 núcleos físicos.
