# Trabajo Práctico Número 6

Arquitectura de Computadoras 13/05/2026

---

**Instrucciones**

- Los ejercicios propuestos en el presente trabajo práctico pueden ser resueltos en forma individual o grupal.
- El planteo de la solución debe realizarse basándose en lo aprendido en las clases teóricas.
- Puede utilizar las clases de consulta para consultar ejercicios de enunciado y verificar la validez de la solución obtenida.
- Posteriormente, se tomará una evaluación con ejercicios similares a los de este práctico, la cual deberá ser resuelta en forma individual. La fecha de evaluación se encuentra en el sitio web de la materia.

---

**Conceptos involucrados**

- Ley de Amdahl para multiprocesamiento. Límite a la aceleración.
- Paralelismo de una tarea.
- Escalabilidad en función del número de procesadores y en función del tamaño del conjunto de datos.
- Escalabilidad fija en el tiempo.
- Comparación multicore versus multithreading.

---

**Problemas propuestos**

1) Dado un programa en el cual el 50 % de las tareas puede reescribirse para aprovechar mejor su paralelismo, el cual se ejecuta en un único procesador, se pide:
   a) Calcule la aceleración que se obtendría si se ejecuta el mismo programa en 4 procesadores.
   b) Calcule nuevamente la aceleración, pero suponiendo que el porcentaje de tareas paralelizables es de 90 %.
   c) ¿Cuál debería ser el porcentaje de tareas paralelizables para que la aceleración sea de 3.8?

2) *[Recomendado]* El siguiente código es utilizado para obtener la suma total de un vector A de 100 elementos:

```
r = A(0);
for(i = 1; i < 100; i++) {
    r = r + A(i)
}
```

   a) Si este programa se ejecuta en un procesador en el que cada suma demora 1 ciclo, calcule cuántos ciclos demorará la ejecución del mismo.
   b) Si se reescribe el código para que se ejecute en dos procesadores como el descrito, ¿cuál será la aceleración obtenida?
   c) Si ahora se reescribe el código para que se ejecute en ocho procesadores como el descrito, ¿cuál será la aceleración obtenida?
   d) ¿Cuál será la máxima aceleración que se pueda obtener? ¿Con cuántos procesadores se alcanza este valor? Saque conclusiones sobre la escalabilidad de la tarea en función del número de procesadores.
   e) Calcule la eficiencia de los multiprocesadores para cada uno de los cuatro casos anteriores.

3) Repita el ejercicio anterior, pero esta vez considerando que el vector posee 1000 elementos. Saque conclusiones sobre la escalabilidad de la tarea en función del tamaño del conjunto de datos.

4) En el siguiente código escrito en lenguaje ensamblador RV32I se muestra una implementación de una búsqueda secuencial de un dato en un vector.

```
main:     li a0, 0              # Resultado de la busqueda, no se encontro el valor
          lw a1, 0(t0)          # Valor a buscar en el vector
          la t0, buscado        # Puntero a la variable con el valor a buscar
          la a2, vector         # Puntero al inicio del vector
```

```
          la t0, cantidad       # Puntero a la variable con la cantidad de elementos
          lw a3, 0(t0)          # Cantidad de elementos en el vector

lazo:     beq zero, a3, final   # Si la cantidad es cero terminar el lazo
          lw t0, 0(a2)          # Carga del elemento del vector
          addi a2, a2, 4        # Incremento del puntero del vector
          beq a1, t0, salir     # Si el elemento es igual se encontro el valor buscado
          addi a3, a3, -1       # Decremento de la cantidad de elementos
          j lazo
salir:    li a0, 1              # Resultado de la busqueda, se encontro el valor
final:    j final
```

Este programa se reescribe para ejecutarlo como dos hilos en un procesador multithreading.

```
main:     la a0, resultado     # Puntero a la variable con el resultado de la busqueda
          sw zero, 0(a0)       # Resultado de la busqueda, no se encontro el valor
          la t0, buscado       # Puntero a la variable con el valor a buscar
          lw a1, 0(t0)         # Valor a buscar en el vector
          la a2, vector        # Puntero al inicio del vector
          la t0, cantidad      # Puntero a la variable con la cantidad de elementos
          lw a3, 0(t0)         # Cantidad de elementos en el vector
          srli a5, a3, 1       # Cantidad de elementos para el segundo hilo
          sub a3, a3, a5       # Cantidad de elementos para el primer hilo
          add a4, a2, a3       # Direccion del primer elemento para el segundo hilo

          # Comienzan a ejecutarse hilo1 e hilo2 concurrentemente

hilo1:    beq zero, a3, final1  # Si la cantidad es cero terminar el lazo
          lw t0, 0(a2)          # Carga del elemento del vector
          beq a1, t0, salir1    # Si el elemento es igual se encontro el valor buscado
          addi a2, a2, 4        # Incremento del puntero del vector
          lw t0, 0(a0)          # Cargar el valor actual del resultado
          addi a3, a3, -1       # Decremento de la cantidad de elementos
          beq zero, t0, hilo1   # Repetir el lazo si no se encontro el resultado
salir1:   li t0, 1              # Resultado de la busqueda, se encontro el valor
          sw t0, 0(a0)          # Actualizar la variable compartida con el resultado
final1:   j final1

hilo2:    beq zero, a5, final2  # Si la cantidad es cero terminar el lazo
          lw t0, 0(a4)          # Carga del elemento del vector
          beq a1, t0, salir2    # Si el elemento es igual se encontro el valor buscado
          addi a4, a4, 4        # Incremento del puntero del vector
          lw t0, 0(a0)          # Cargar el valor actual del resultado
          addi a5, a5, -1       # Decremento de la cantidad de elementos
          beq zero, t0, hilo2   # Repetir el lazo si no se encontro el resultado
salir2:   li t0, 1              # Resultado de la busqueda, se encontro el valor
          sw t0, 0(a0)          # Actualizar la variable compartida con el resultado
final2:   j final2
```

A partir de estos códigos, los cuales se ejecutan en un procesador con CPI igual a 1, y sabiendo que operan sobre un vector de 10 elementos, se pide que:

a) Calcule el tiempo de ejecución de la versión original en el peor caso (cuando que el elemento no está en vector).
b) Calcule el tiempo de ejecución de la versión multithreading en el peor caso (cuando que el elemento no está en vector).
c) Calcule la aceleración obtenida con la versión multithreading respecto de la versión original.
d) Recalcule los valores del apartado anterior para 1000 elementos.
e) Explique brevemente por qué, en la versión multithreading, la variable resultado está en memoria y no en un registro.

5) *[Recomendado]* Según el ranking de www.top500.org, la computadora más rápida del mundo a junio de 2023 es la Frontier, ubicada en el Laboratorio Nacional de Oak Ridge, en Estados Unidos. La computadora está compuesta por 72 racks, cada uno con 128 nodos, y donde cada nodo tiene 1 procesador AMD EPYC 7003 junto con 4 aceleradores (procesadores gráficos) AMD Instinct MI250X. Los procesadores EPYC 7003 tienen 64 núcleos, donde cada núcleo tiene una frecuencia de 2 GHz y puede ejecutar hasta 10 instrucciones de punto flotante por ciclo. Los

procesadores MI250X tienen 220 núcleos de cómputo, donde cada núcleo tiene una frecuencia de 1,62 GHz y puede ejecutar hasta 128 instrucciones de punto flotante por ciclo.

a) ¿Cuántos núcleos posee en total esta computadora?
b) ¿Cuánto es la performance pico de esta computadora, medida en TFLOPS? Para el cálculo considere únicamente a los aceleradores gráficos.
c) En esta computadora, quiere ejecutarse un benchmark que consiste en realizar un determinado procesamiento a un conjunto de datos, donde cada dato es independiente del resto. Si el procesamiento consiste en 4 operaciones de punto flotante, y la cantidad de datos independientes es de $10^{20}$, ¿cuánto tiempo demorará el benchmark en ejecutarse, suponiendo que no hay overhead por sincronización?
d) Utilizando información del ranking mencionado, ¿cuál es la segunda supercomputadora más rápida del mundo?. ¿Cuánto más rápida es Frontier?
e) Según el mismo ranking, ¿cuánta energía consumen cada una de estas supercomputadoras? ¿Cuál de las dos es más eficiente, considerando performance por watt?

**Ejercicio a Desarrollar en Clase Práctica**

**Problema 3 del Segundo Parcial de 2018**

1) Suponga que usted es contratado por una empresa para realizar su pasantía, y su trabajo consiste en paralelizar un viejo programa de manera que pueda ejecutarse más velozmente en un procesador moderno con varios núcleos.
   a) Usted descubre que existe una actualización de estado global que ralentiza todo el programa. En particular, usted puede medir que el programa ejecuta el 90 % de su trabajo en la sección paralelizable, mientras que el 10 % restante es ejecutado de manera secuencial. ¿Cuál sería la máxima aceleración que se podría obtener si el procesador tuviese infinitos núcleos?
   b) Con las consideraciones del apartado anterior, ¿cuál sería la aceleración al utilizar un procesador con 16 núcleos?
   c) Como en su empresa no están satisfechos con la aceleración obtenida con los 16 núcleos, se le ocurre probar con un procesador heterogéneo, compuesto por 12 núcleos como los del apartado anterior, más un núcleo más grande. Este núcleo más grande es capaz de ejecutar la porción serial del código y obtener una aceleración igual a 2 en la misma. Los otros 12 núcleos se encargarían de ejecutar la porción paralelizable del código. ¿Es buena su sugerencia de probar con este núcleo? ¿Cuánto es la aceleración que se obtiene de esta manera?
   d) El gerente de su empresa aún no está satisfecho con la aceleración obtenida, por lo que usted redobla sus esfuerzos y consigue optimizar la parte serial del código para que ocupe solamente el 4 % del tiempo, mientras que el 96 % restante es paralelizable. ¿Cuánto sería la aceleración en este caso, utilizando el procesador heterogéneo del apartado anterior? ¿Y cuánto sería la aceleración considerando el procesador con los 16 núcleos iguales?

**Soluciones propuestas**

1)
   a) A = 1,60.
   b) A = 3,07.
   c) El porcentaje de tareas paralelizables debe ser del 98,2 %.

2)
   a) La ejecución demorará 99 ciclos.
   b) La aceleración sería A = 1.98, casi lineal.
   c) La aceleración sería A = 6.6, por lo que vemos que deja de ser lineal.
   d) La máxima aceleración posible es A = 14.14, y la misma se logra con 50 procesadores.
   e) $E_{1P}$ = 100 %, $E_{2P}$ = 99 %, $E_{8P}$ = 83 % y $E_{50P}$ = 28 %.

3)
   a) La ejecución demorará 999 ciclos.
   b) La aceleración sería A = 1.998, casi lineal.
   c) La aceleración sería A = 7.866, sigue siendo bastante lineal.
   d) La máxima aceleración posible es A = 99.9, y la misma se logra con 500 procesadores.
   e) $E_{1P}$ = 100 %, $E_{2P}$ = 100 %, $E_{8P}$ = 98 % y $E_{500P}$ = 20 %.

4)
   a) 67T
   b) 46T
   c) 1,46
   d) 6007T, 3511T y 1,71
   e) ...

5)
   a) 8.699.904 núcleos.
   b) 1.681.736 TFLOPS.
   c) 4 minutos, aproximadamente.
   d) Frontier es 3.13 veces más rápida que la segunda, llamada Fugaku.
   e) Frontier consume 22.703 kW y Fugaku 29.899 kW, por lo que Frontier es 4,11 veces más eficiente.