

IMT2112 - Algoritmos Paralelos en Computación Científica

Eficiencia paralela

Elwin van 't Wout

27 de agosto de 2019



PONTIFICIA
UNIVERSIDAD
CATÓLICA
DE CHILE

Facultad de Matemáticas • Escuela de Ingeniería

imc.uc.cl

Clases previas

La arquitectura de un computador

- Multiprocessing
- Paralelismo de nivel de instrucción
- Jerarquía de memoria
- Localidad de datos

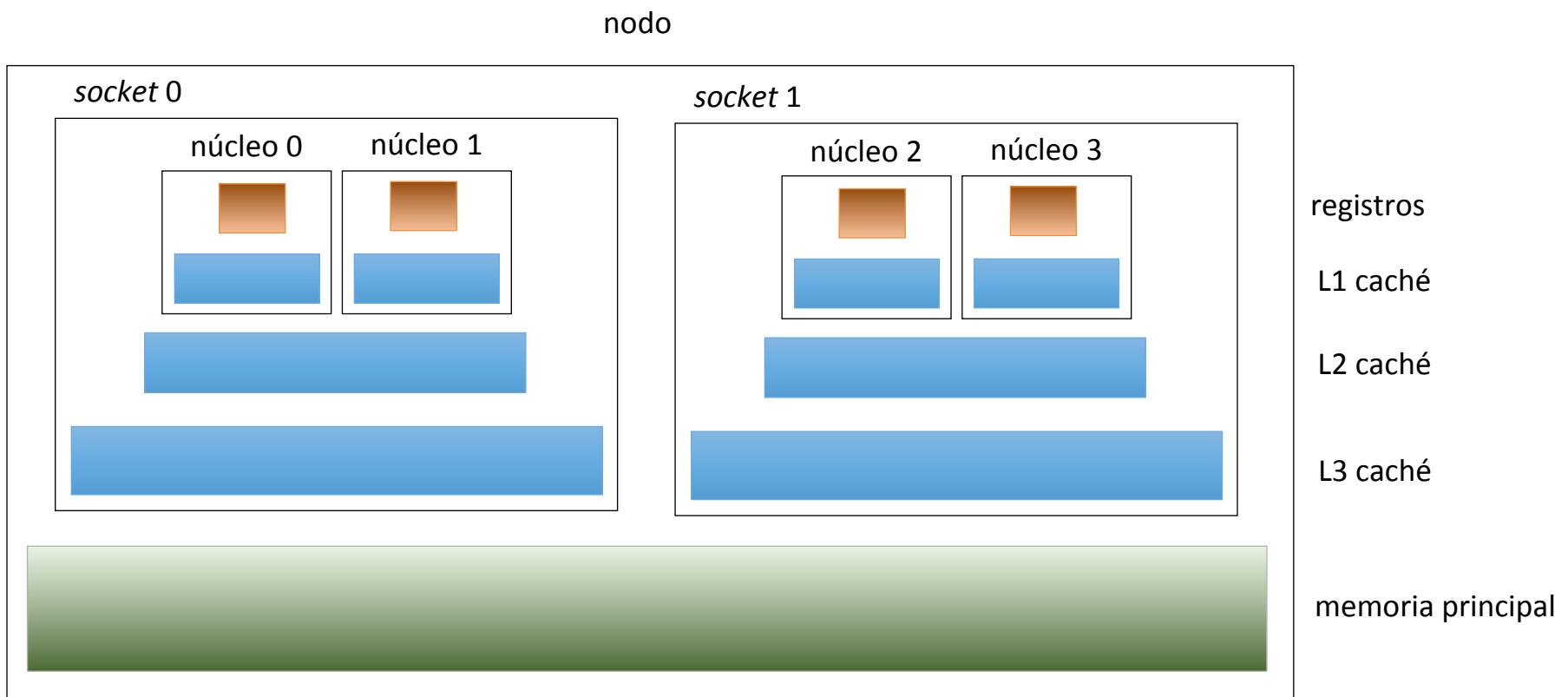
Agenda

- ¿Como cuantificar el rendimiento paralelo de un algoritmo?

La computación paralela

Sección 2.1 del libro de Eijkhout

Los procesadores *multi-core*



Los procesadores *multi-core*

- Núcleos físicos
 - el número de dispositivos de cálculo en el *chip*
- Núcleos virtuales o lógicos
 - el número de distintas secuencias de instrucciones
 - si es mayor que el físico se llama *hyperthreading*

El paralelismo en algoritmos

- Paralelismo de datos (*data parallelism*)
 - la misma operación aplicado a distintos datos
- Paralelismo funcional (*functional parallelism*)
 - ejecutar distintas operaciones independientes en paralelo
- Paralelismo de tareas (*task parallelism*)
 - ejecutar distintas partes del código en paralelo

Cuantificar el paralelismo

Sección 2.2 del libro de Eijkhout

Speedup y eficiencia

- El *speedup* (aceleración) se define como $S_p = T_1/T_p$ con T_p el tiempo de ejecución de los p procesadores
- La eficiencia se define como $E_p = S_p/p$
- Estas son medidas crudas
 - capacidad de almacenamiento
 - transferencia de datos
 - arquitectura de computadores
 - optimizaciones algorítmicas

Dependencia de datos

- El paralelismo está restringido por instrucciones dependientes sobre elementos de datos
- El *camino crítico*:
 - La secuencia de instrucciones completamente dependiente más larga
 - Esto determina el tiempo de ejecución cuando hay disponible una cantidad ilimitada de procesadores
- Poco realista porque se ignora el *overhead* de comunicación

Ley de Amdahl

- Para T_p el tiempo de ejecución de los p procesadores y F_s y F_p la fracción secuencial y paralela, tenemos

$$T_p = T_1(F_s + F_p/p) + T_{\text{comm}}$$

donde T_{comm} denota el tiempo de comunicación

- La aceleración paralela es por lo tanto

$$S_p \leq \frac{1}{F_s + F_p/p}$$

Ley de Gustafson

- En la ley de Amdahl, asumimos que la fracción secuencial del código era fija
- En la práctica, esto depende del tamaño del problema
 - la ‘fracción secuencial’ refiera a la cantidad de instrucciones de bajo nivel
- Ley de Amdahl:
 - código para 1 procesador → tiempo en p procesadores
- Ley de Gustafson:
 - código para p procesador → tiempo en 1 procesador

Leyes de Amdahl y Gustafson

- Amdahl: $T_p = T_1(f_1 + (1 - f_1)/p) + T_{\text{comm}}$
- Gustafson: $T_1 = T_p(f_p + (1 - f_p)p) + T_{\text{comm}}$
- Note la diferencia en eficiencia paralela
 - Amdahl: $T_1/T_p = \mathcal{O}(1)$
 - Gustafson: $T_1/T_p = \mathcal{O}(p)$
- Gustafson = Amdahl si $f_p = \frac{f_1}{f_1 + (1 - f_1)/p}$
- Aquí, f_p denota la fracción secuencial para un código ejecutado en p procesadores

Leyes de Amdahl y Gustafson

- Los problemas a gran escala suelen tener más posibilidades de parallelizar
- La parte “escalable” del código a menudo es más fácil de parallelizar
 - El código secuencial incluye: entrada, salida
 - El código paralelo incluye: álgebra lineal, sortear

Escabilidad

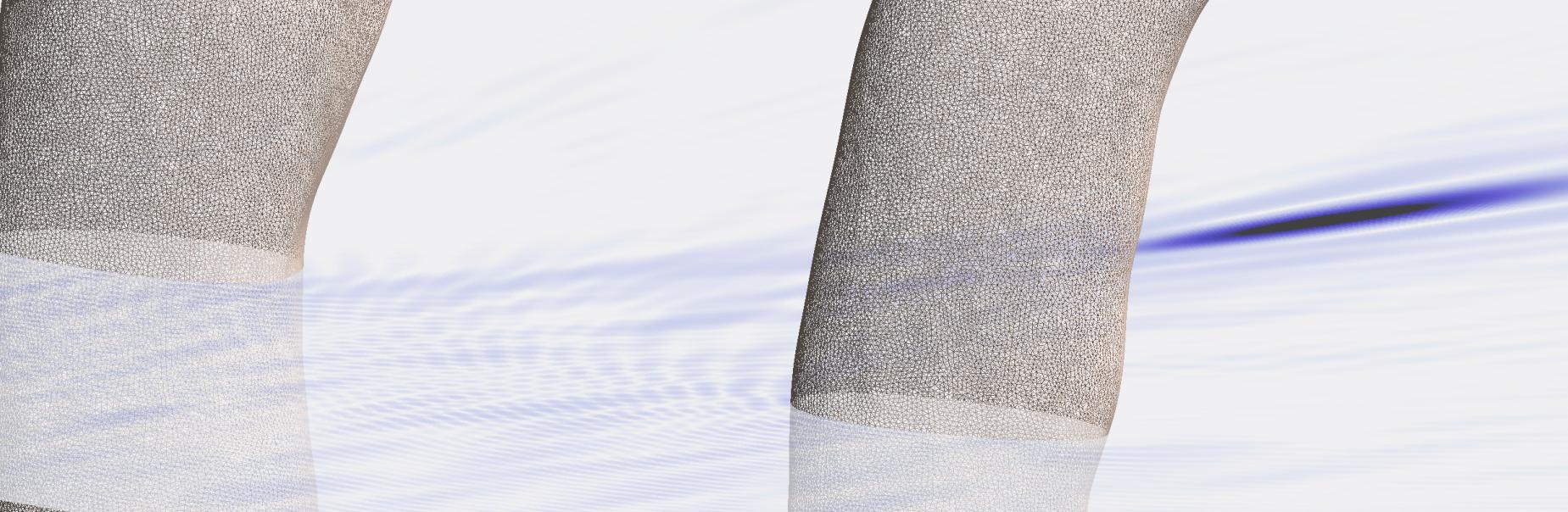
- Escalabilidad fuerte (*strong scaling*)
 - aumentar el número de procesadores para un código dado
 - la referencia es un código secuencial o un código paralelo en una pequeña cantidad de procesadores
- Escalabilidad débil (*weak scaling*)
 - aumentar la cantidad de datos o instrucciones junto al número de procesadores
 - esto permite una mayor eficiencia paralela

Resumen

- *Speedup* y eficiencia paralela
- Leyes de Amdahl y Gustafson

Clase siguiente

- Acceso a memoria paralela
- Granularidad
- Balanceo de carga



IMT2112 - Algoritmos Paralelos en Computación Científica

Eficiencia paralela

Elwin van 't Wout

27 de agosto de 2019



PONTIFICIA
UNIVERSIDAD
CATÓLICA
DE CHILE

Facultad de Matemáticas • Escuela de Ingeniería

imc.uc.cl