

BAB 14

Arsitektur Paralel dan Multiprosesor

IF221101 • Informatika • 3 SKS

Capaian Bab Ini

- Mengapa pemrosesan paralel diperlukan
- Concurrency vs parallelism
- Paralelisme instruksi, data, tugas, thread
- Klasifikasi Flynn
- Multicore, multiprosesor, shared/distributed memory
- Sinkronisasi, race condition, critical section
- Speedup, efficiency, scalability, Hukum Amdahl
- Load balancing dan overhead

Concurrency vs Parallelism

Konsep	Fokus	Contoh
Concurrency	Pengelolaan beberapa aktivitas	Task bergantian/overlap
Parallelism	Eksekusi serentak	Dua core mengolah data bersamaan
Hubungan	Dapat muncul bersama	Sistem multicore menjalankan thread paralel

Parallelism menekankan eksekusi bersamaan; concurrency menekankan struktur dan pengelolaan aktivitas.

Instruksi, Data, Tugas, Thread

Tingkat	Ide Utama	Contoh
Instruksi	Beberapa instruksi overlap	Pipeline CPU
Data	Operasi sama pada banyak data	Vector/SIMD
Tugas	Tugas berbeda berjalan bersama	Layanan paralel
Thread	Alur eksekusi dalam aplikasi	Multithreading

Klasifikasi Arsitektur Paralel

Kategori	Instruksi	Data	Gambaran
SISD	Satu	Satu	Komputasi serial
SIMD	Satu	Banyak	Operasi sama pada banyak data
MISD	Banyak	Satu	Jarang digunakan secara umum
MIMD	Banyak	Banyak	Multicore/multiprosesor umum

Flynn adalah kerangka konseptual berdasarkan jumlah aliran instruksi dan data.

Shared vs Distributed Memory

Aspek	Shared Memory	Distributed Memory
Lokasi data	Ruang memori bersama	Memori lokal tiap node
Komunikasi	Akses struktur bersama	Pesan/jaringan antar node
Tantangan	Sinkronisasi dan kontensi	Komunikasi dan pembagian data
Contoh	SMP, multicore	Cluster

Race Condition dan Critical Section

Race condition terjadi ketika hasil bergantung pada urutan/waktu operasi bersamaan. Critical section adalah kode yang mengakses sumber daya bersama.

Masalah	Pengendalian
Race condition	Lock, mutex, semaphore, atomic operation
Critical section	Batasi akses satu thread pada satu waktu
Cache coherence	Pastikan salinan cache tetap sesuai

Speedup, Efficiency, Scalability

Ukuran	Rumus	Contoh
Speedup	$T_{\text{serial}}/T_{\text{parallel}}$	$100/25 = 4$
Efficiency	$\text{Speedup}/P$	$6/8 = 75\%$
Scalability	Kinerja saat resource/workload bertambah	Tidak selalu linear

Penambahan core tidak otomatis memberi speedup linear: serial, komunikasi, dan imbalance tetap ada.

Batas Percepatan Teoritis

Jika P = proporsi yang dapat diparalelkan dan N = jumlah prosesor:

$$\text{Speedup} = 1 / ((1-P) + P/N)$$

Contoh $P=0,90$ dan $N=10 \rightarrow 1/(0,10+0,09)=5,26$

Bagian serial membatasi speedup maksimum meskipun prosesor terus ditambah.

Load Balancing dan Granularity

Konsep	Makna	Risiko
Load balancing	Bagi beban relatif seimbang	Core menganggur jika tidak seimbang
Granularity	Ukuran pekerjaan per unit	Terlalu kecil = overhead tinggi
Overhead	Biaya komunikasi/koordinasi	Mengurangi speedup

Perkalian Matriks Paralel

Matriks $N \times N$ dapat dibagi berdasarkan baris, kolom, atau blok. Setiap worker menghitung bagian hasil, lalu hasil digabungkan.

Pembagian	Kelebihan	Catatan
Baris	Sederhana	Butuh akses kolom
Kolom	Mudah dipisah	Komunikasi hasil
Blok	Locality lebih baik	Pembagian lebih kompleks

Analisis Hukum Amdahl

Prosesor	P=80% (ideal)	Insight
2	$1/(0,2+0,8/2)=1,67$	Speedup awal besar
4	2,50	Tambahan mulai turun
8	3,33	Diminishing returns
16	4,00	Batas maksimum = 5

Semakin banyak unit, tambahan percepatan makin kecil karena bagian serial 20%.

Race Condition

Dua thread menaikkan variabel bersama. Increment = baca → hitung → tulis. Tanpa sinkronisasi, operasi bisa tumpang tindih dan hasil akhir salah.

Tanpa Sinkronisasi	Dengan Critical Section
Lost update	Satu thread menimpa hasil lain
Hasil nondeterministik	Urutan timing menentukan output
Solusi	Lock/mutex/atomic increment

Tugas dan Proyek Mini

- Identifikasi data/task/thread-level parallelism
- Klasifikasikan skenario SISD/SIMD/MIMD
- Bagi pekerjaan ke 2, 4, 8 worker
- Hitung speedup dan efficiency
- Analisis shared vs distributed memory
- Simulasi race condition dan critical section

Proyek: Parallel Computing and Performance Simulator.

Poin Utama Bab 14

- Parallelism membagi pekerjaan untuk eksekusi bersamaan
- Concurrency dan parallelism punya fokus berbeda
- Flynn: SISD, SIMD, MISD, MIMD
- Shared memory perlu sinkronisasi; distributed perlu komunikasi
- Race condition muncul pada resource bersama
- Speedup, efficiency, scalability mengukur kinerja

SELANJUTNYA: BAB 15: Sistem Terdistribusi dan Cloud Computing

- Load imbalance dan overhead mengurangi manfaat core

Daftar Pustaka

- Patterson, D. A., & Hennessy, J. L. Computer Organization and Design. Morgan Kaufmann.
- Hennessy, J. L., & Patterson, D. A. Computer Architecture: A Quantitative Approach. Morgan Kaufmann.
- Stallings, W. Computer Organization and Architecture. Pearson.
- Grama, A., Gupta, A., Karypis, G., & Kumar, V. Introduction to Parallel Computing. Addison-Wesley.
- Null, L., & Lobur, J. The Essentials of Computer Organization and Architecture. Jones & Bartlett Learning.
- RPS Arsitektur Komputer IF221101.