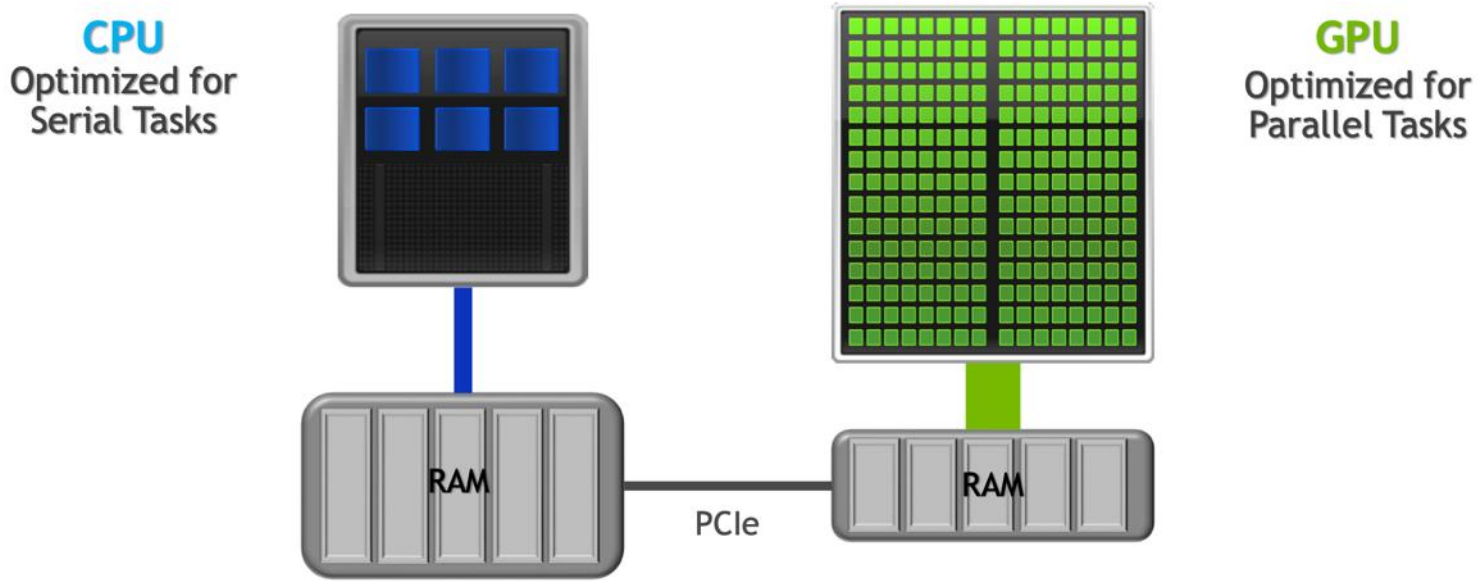
Insper

Supercomputação

Recaptulando...

- Diferenciar dispositivos de latência (CPUs) e de throughput (GPUs)
- Compreender o layout de memória e transferência de dados em sistemas heterogêneos (CPU ⇔ GPU)
- Compilar primeiros programas na GPU

CPUs e GPUs


CPU
Optimized for
Serial Tasks
GPU
Optimized for
Parallel Tasks
RAM
PCIe
RAM

Speed vs Throughput

Speed

Throughput

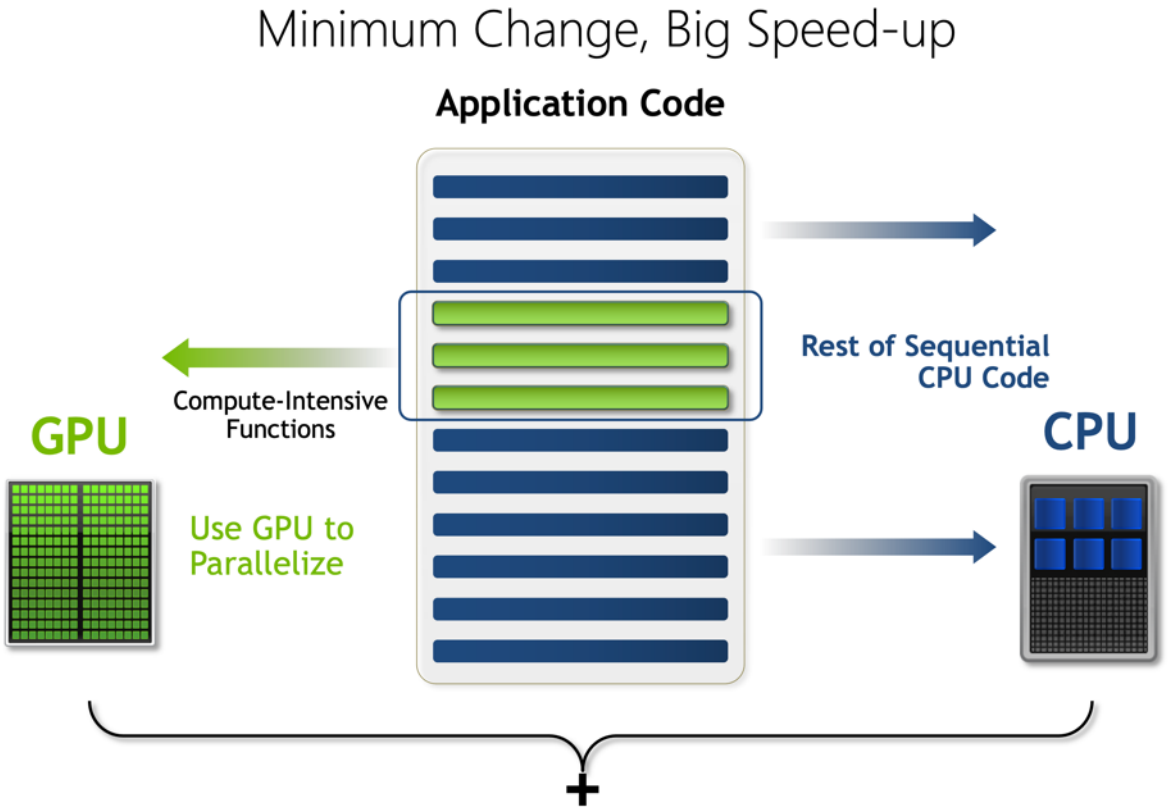
Which is better depends on your needs...

CPU vs GPU

- CPUs para partes sequenciais onde uma latência mínima é importante
  - CPUs podem ser 10X mais rápidas que GPUs para código sequencial

- GPUs para partes paralelas onde a taxa de transferência(throughput) bate a latência menor.
  - GPUs podem ser 10X mais rápidas que as CPUs para código paralelo

CPU vs GPU


Minimum Change, Big Speed-up
Application Code
Compute-Intensive Functions
Rest of Sequential CPU Code
GPU
CPU
Use GPU to Parallelize

Programando para GPU

- Compilador especial: nvcc
- Endereçamento de memória separado
    - Dados precisam ser copiados de/para GPU
    - Isto leva tempo
- Funções especiais (kernels) para rodar na GPU

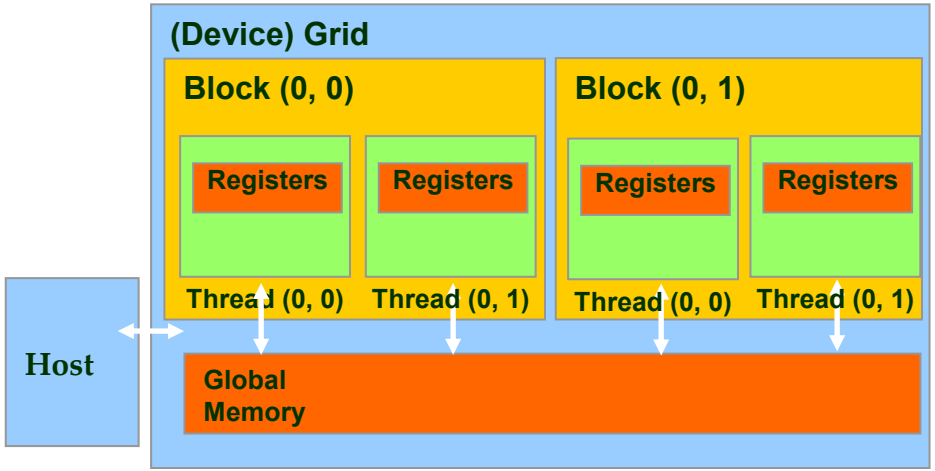
Memória em GPUs

Código da GPU (device) pode:

- Cada thread ler e escrever nos registradores
- Ler e escrever na memória global

Código da CPU (host) pode:

- Transferir dados de e para memória global

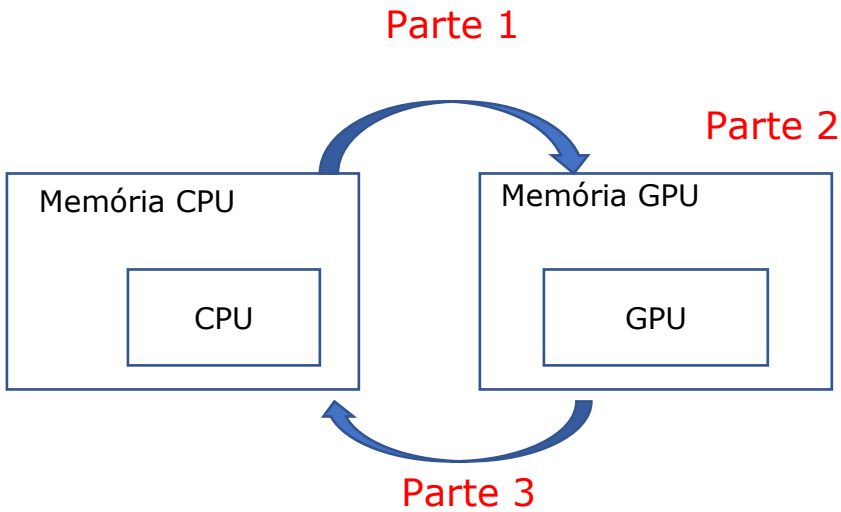


Fluxo dos programas

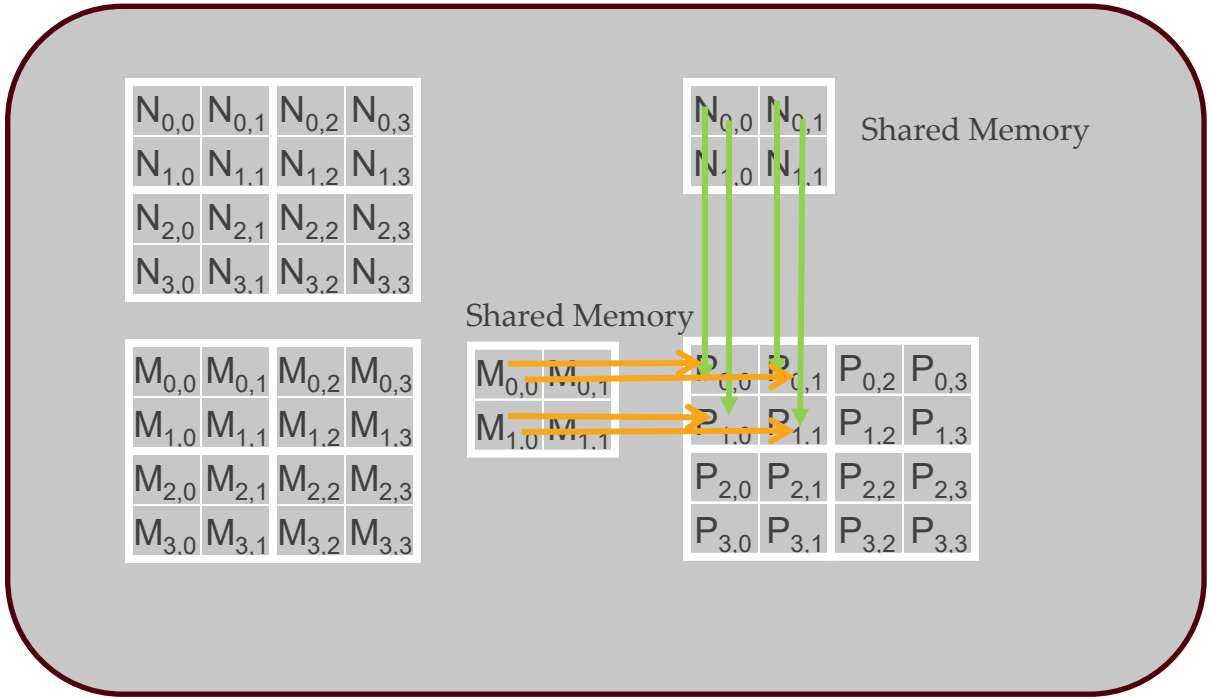
Parte 1: copia dados CPU → GPU

Parte 2: processa dados na GPU

Parte 3: copia resultados GPU → CPU

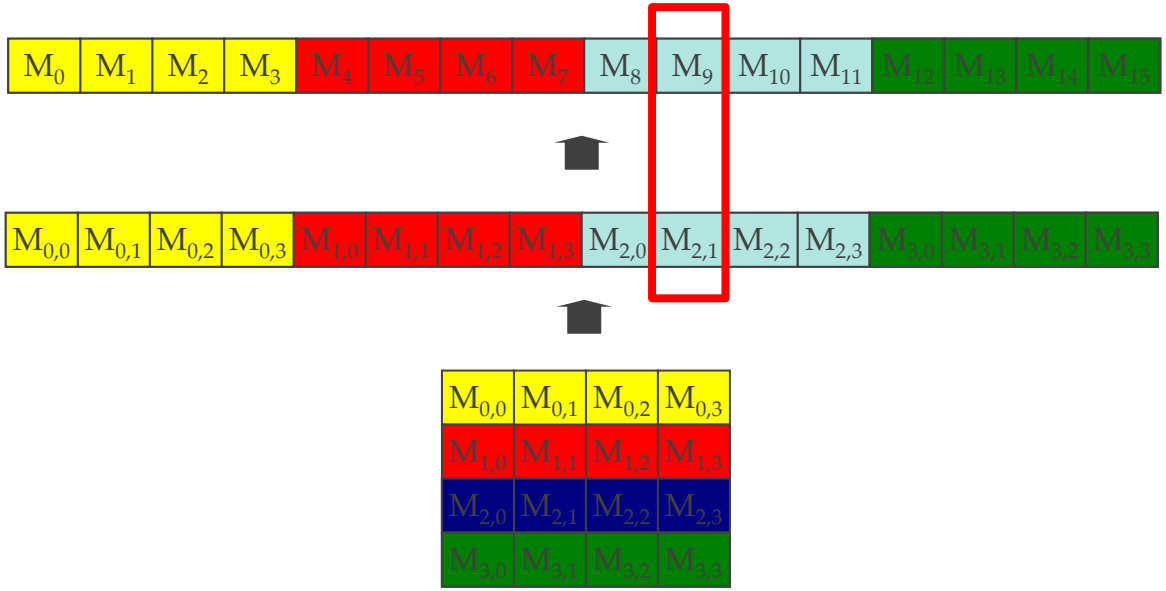


Acessos em memória



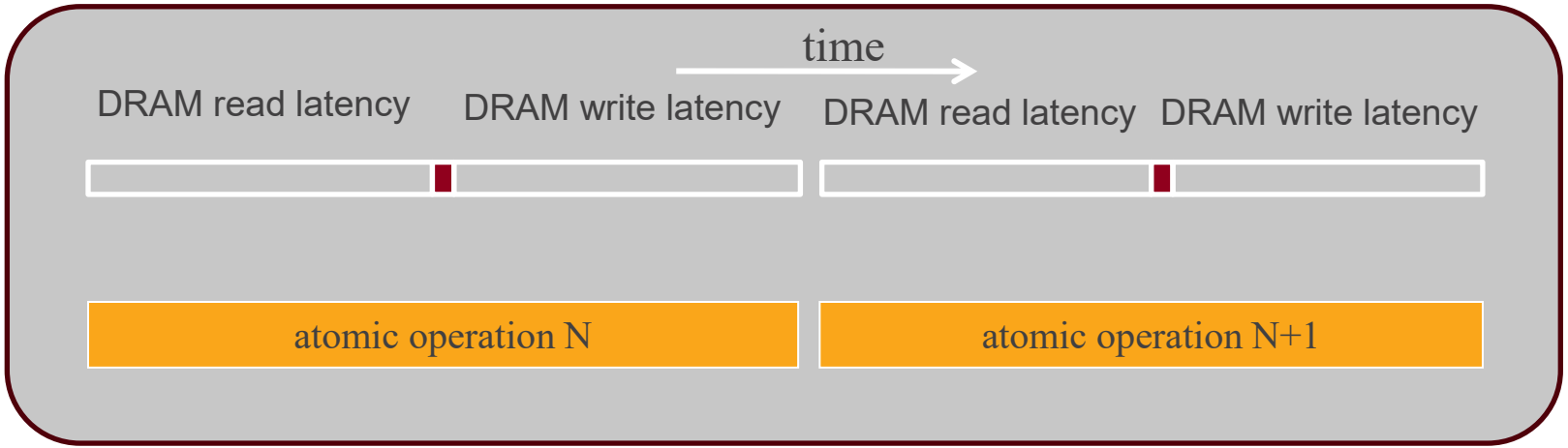
Encontrando o dado

Posição do dado = Linha x Elementos + Coluna = 2 x 4 + 1 = 9



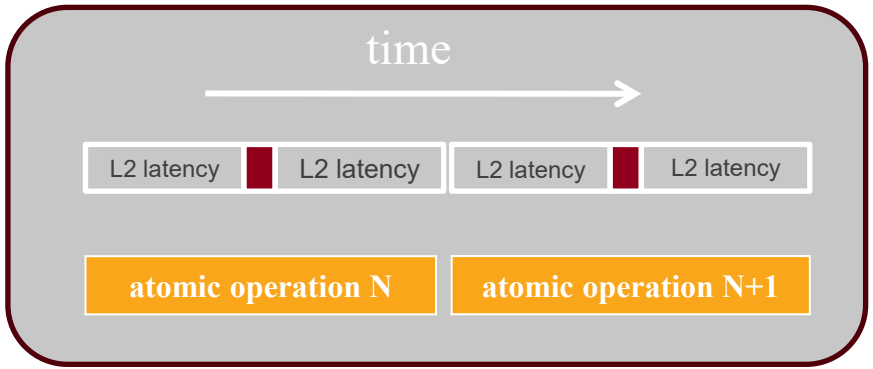
Operações atômicas na DRAM

Todas as operações atômicas são serializadas



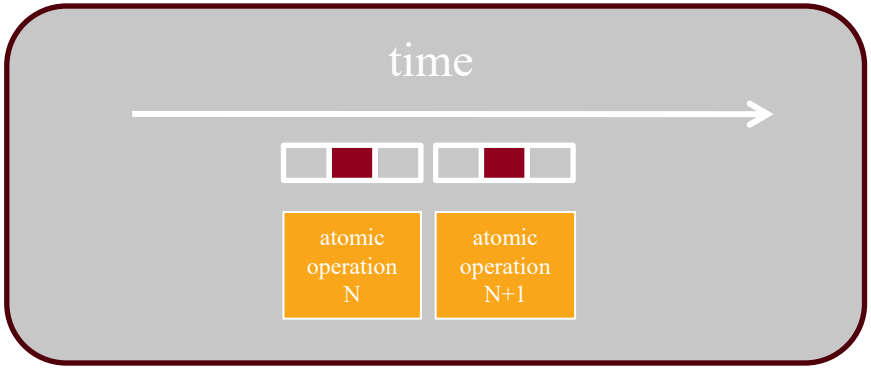
Operações atômicas na Cache

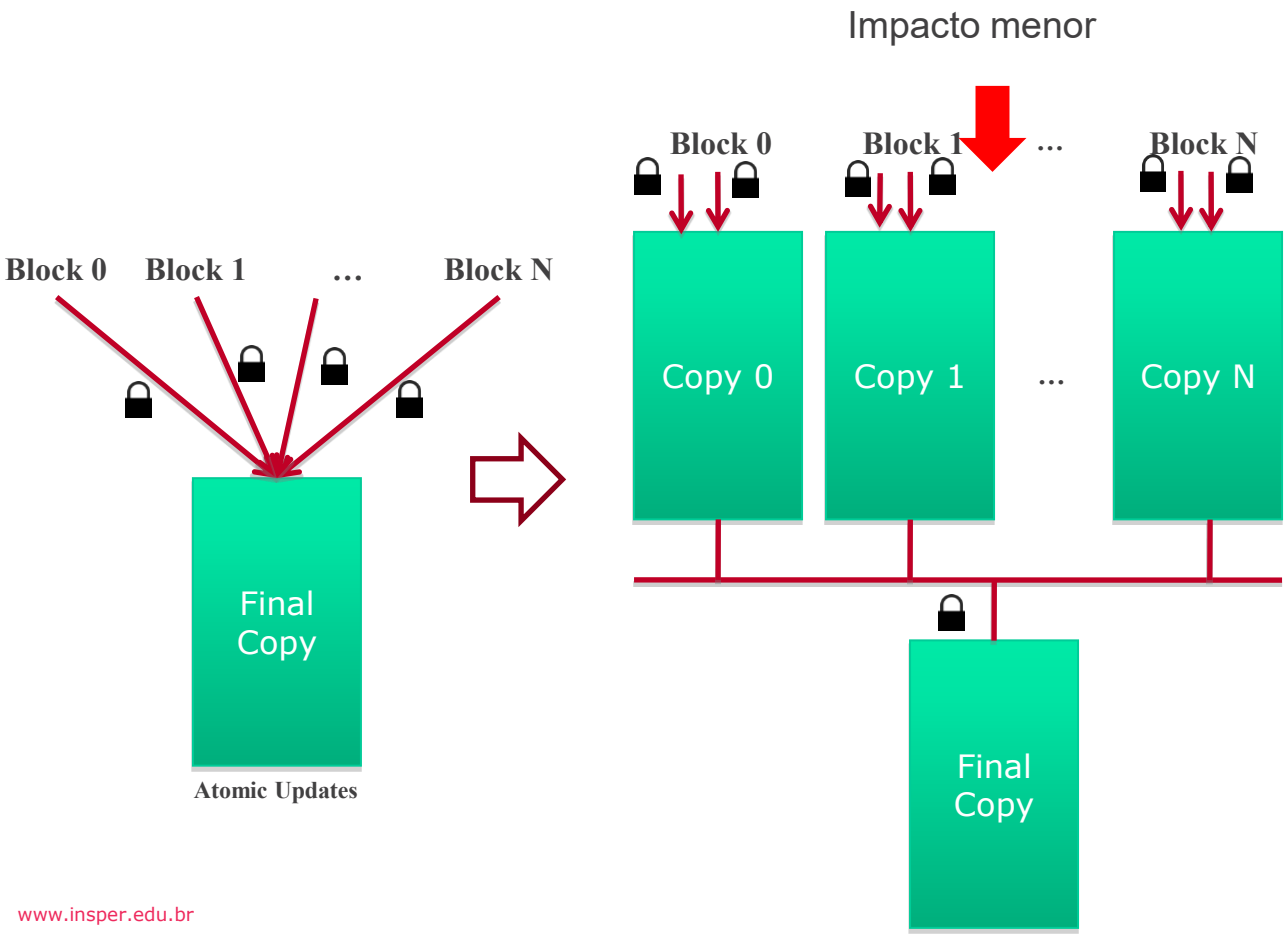
- Latência media, aproximadamente 1/10 em comparação a DRAM
- Compartilhado entre todos os blocos



Operações atômicas na Shared Memory

- Latência muito pequena
  - Privado para cada bloco de thread
  - Menor impacto global




Impacto menor
Block 0
Block 1
...
Block N
Final Copy
Atomic Updates
Block 0
Block 1
...
Block N
Copy 0
Copy 1
...
Copy N
Final Copy