

Pure Attention

Deconstructing the Magic: A White-Box Engineering Approach

Sincari Sebastian-George
Andrei Cristian-David

Universitatea din București - Facultatea de Matematică și Informatică
Specializarea Informatică

Februarie, 2026

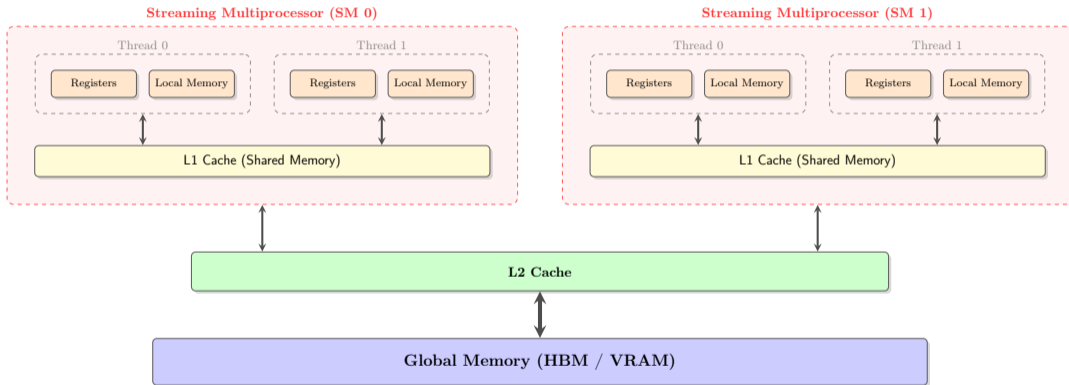
Problema Generală

- Librăriile moderne de Deep Learning transformă funcționalitățile în "Black Box".
- **Motivație:** Demistificarea "magiei" din spatele antrenării (scop educațional).

Contribuție personală (PureAttention)

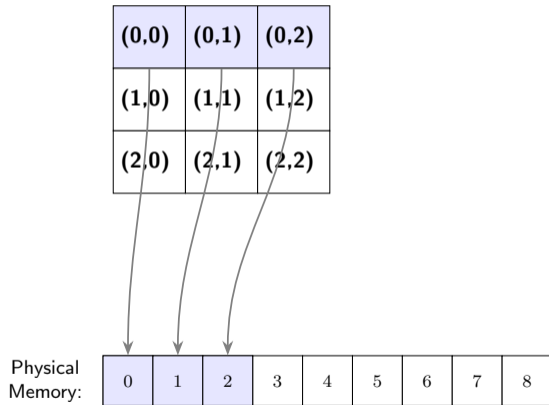
- Implementare *White-box* a mecanismelor specifice Deep Learning.
- Folosire **CUDA** pentru implementările low-level, **C++** pentru abstractizarea funcționalităților și **Python** pentru wrapping.
- Implementarea **Flash Attention V2**.
- Validare matematică vs. soluții analitice și PyTorch.
- Analiza performanței (GFLOPS) și a optimizărilor kernel.

Arhitectura GPU



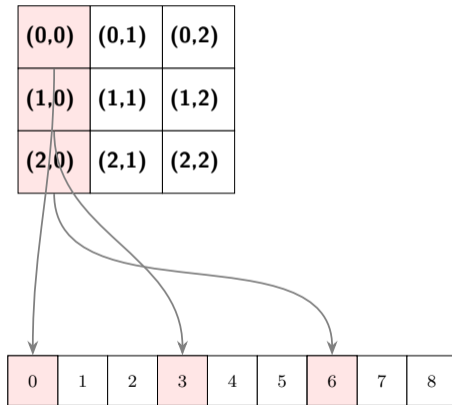
Memory Coalescing

Row-Major
(Coalesced Access)



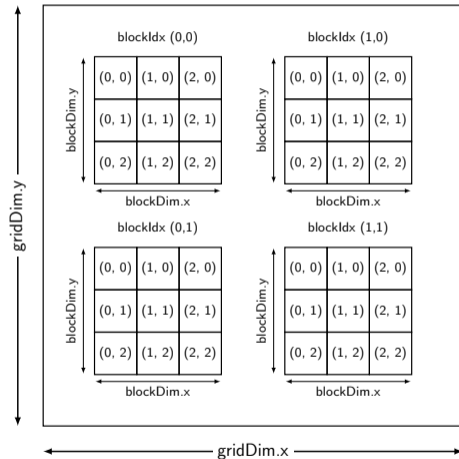
One Memory Transaction

Column-Major
(Strided Access)



Multiple Memory Transactions

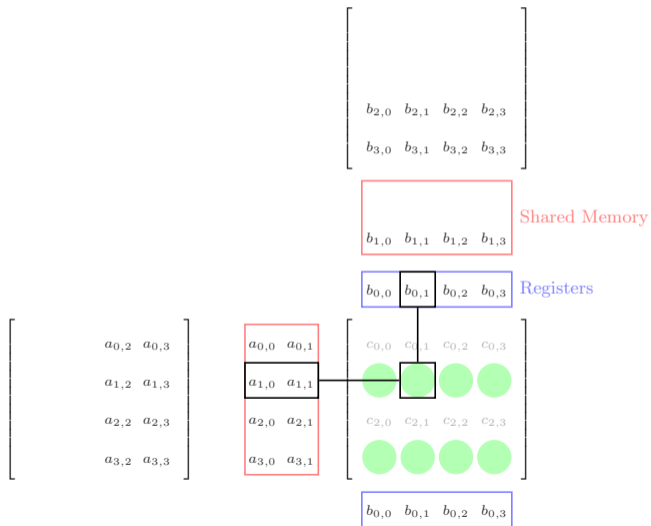
Thread Indexing



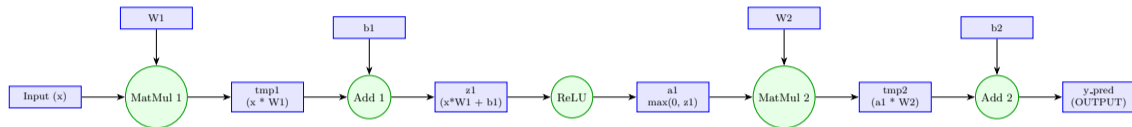
$$x = blockIdx.x \cdot blockDim.x + threadIdx.x \quad (1)$$

$$y = blockIdx.y \cdot blockDim.y + threadIdx.y \quad (2)$$

Tiled Shared Memory & Thread Coarsening

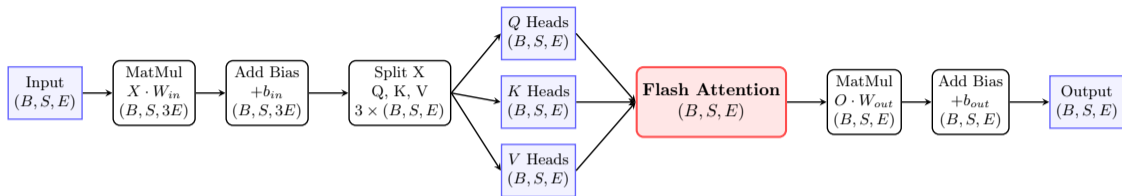


Arhitectura Sistemului & Funcționalități Implementate



- **Funcție de Loss:** Mean Squared Error (MSE)
- **Optimizator:** Adam
- **Funcție de Activare:** ReLU
- **Inițializare Weights:** He Initialization

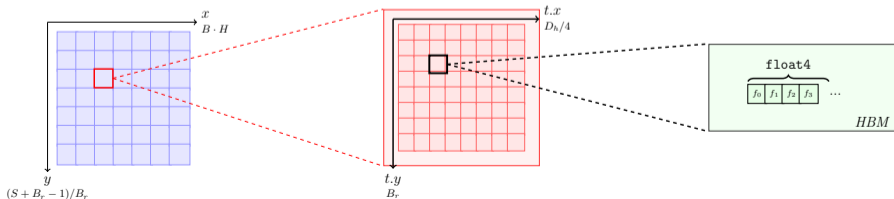
Flash Attention Pipeline



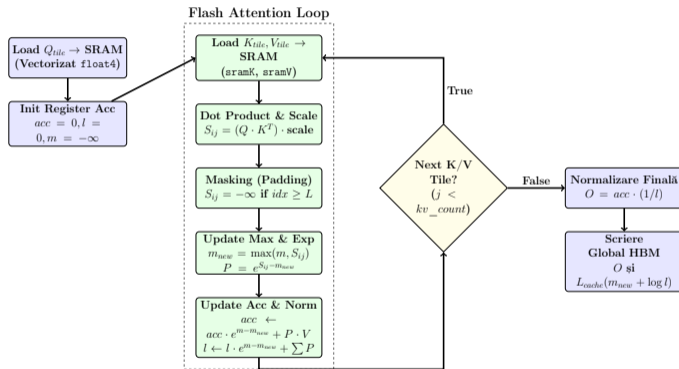
1. CUDA Grid
(GridDim)

2. Thread Block
(BlockDim)

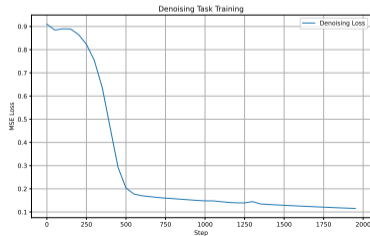
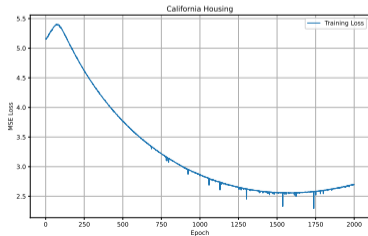
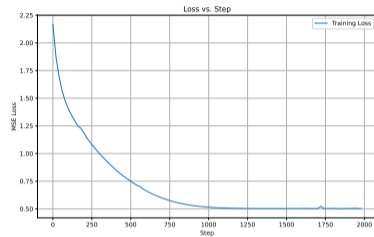
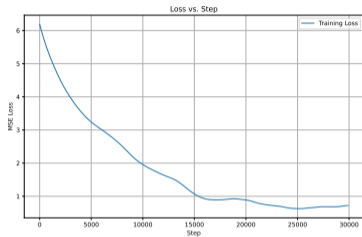
3. Memory Map
(float4)



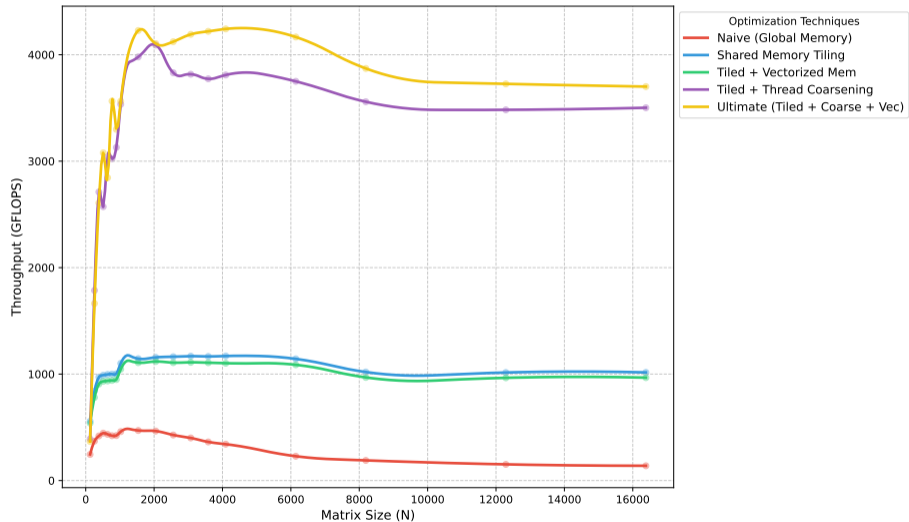
Flash Forward Execution



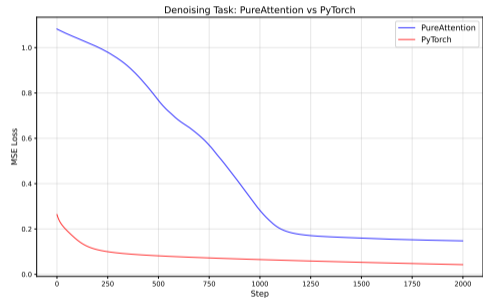
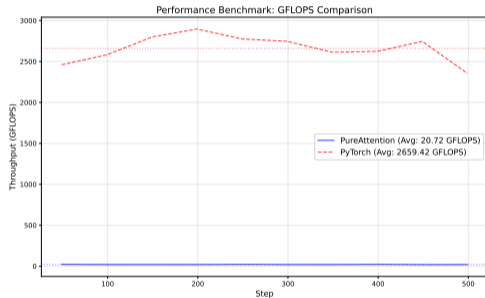
Validare Matematică și Convergență pe Serii de Timp



Performanța Kernel



Comparare cu PyTorch



- **Implementarea unui Caching Allocator:** Prin analiza de performanță am identificat alocarea dinamică a memoriei (apeluri repetitive de `cudaMalloc`) ca fiind un bottleneck major. O strategie de reutilizare a blocurilor de memorie ar reduce latențele semnificativ.
- **Suport pentru Tensor Cores:** Utilizarea instrucțiunilor `nvcuda::wmma` pentru a exploata unitățile matriciale specializate din arhitecturile Volta și Ampere, permițând accelerarea calculelor.
- **Kernel Fusion:** Combinarea operațiilor adiacente din graful de computație (ex: `MatMul + Bias + Activation`) într-un singur kernel pentru a reduce traficul de memorie globală și a crește intensitatea aritmetică.