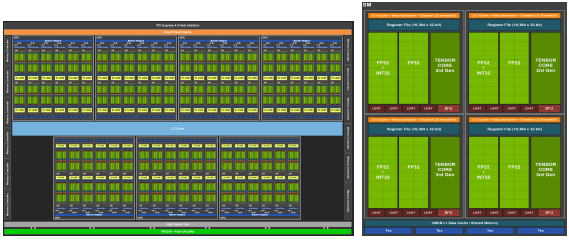
Qualitative and quantitative study of modern GPU synchronization approaches

Ilya Pershin, Vadim Levchenko and Anastasia Perepelkina
Keldysh Institute of Applied Mathematics, Moscow, Russia

September 27, 2021

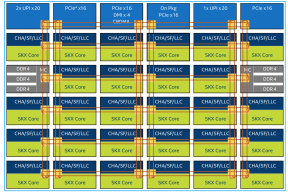
Parallelism levels

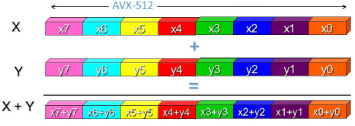
One GPU



$\underline{10^2\ \text{SMs}} \times 128$ CUDA Cores $=$
10^4 FLOP / clock

CPU Cluster





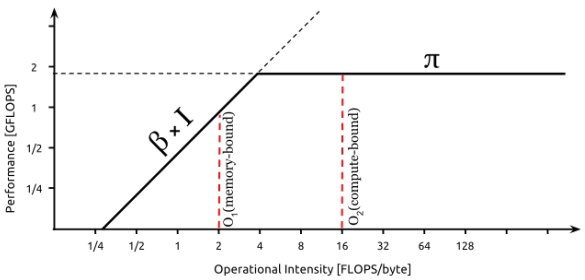
$10^{2 \div 5}$ Nodes $\times$ 32 Cores $\times$ 32 FLOP / Core $=$
$10^{5 \div 8}$ FLOP / clock

High parallelism hardware issues

Memory wall

Why: Computing >> Data transfer

Model: Roofline



Load balancing

Why: Computing heterogeneity, Sync latency

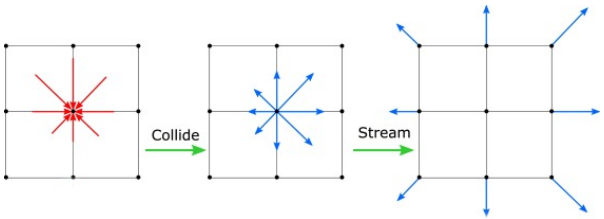
Model: ?? (Spoiler: New Tau-model)

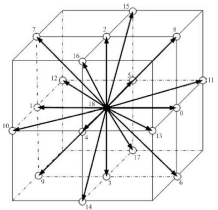
Task:

1. Create high-performance 3D stencil (LBM) code
2. Focus on SMs' synchronization approaches
3. Build a quantitative model

Lattice Boltzmann method (LBM)

- In each cell, fluid flow $\rightarrow \{f_i\}$ with constant $\vec{c}_i$, $i = 1 \dots Q$
- Two-stage stencil scheme
  - Collision: $f_i^*(\vec{x}, t) = \Omega(f_1(\vec{x}, t), \dots, f_Q(\vec{x}, t))$ **Local computing**
  - Streaming: $f_i(\vec{x} + \vec{c}_i, t + 1) = f_i^*(\vec{x}, t)$ **Data transfer**
- Macroscopic parameters: $\rho = \sum_i f_i$, $\rho\vec{u} = \sum_i f_i \vec{c}_i$, etc.
- Specifically, LBM D3Q19 & BGK Ω is used

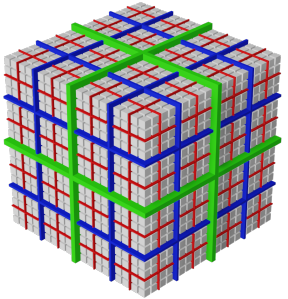


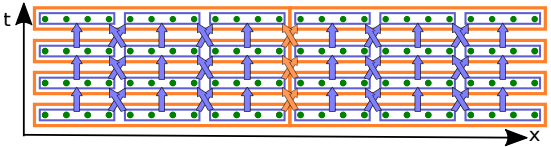


Recursive Domain Decomposition

RDD boosts operational intensity of a stencil code

- SM' register file localization (256 KB per SM)
- 3D mesh → recursive rectangular blocks
- Decomposition level = Parallelism level
- Blocks' data exchange:
  - Threads in a warp → Warp shuffle
  - Warps → Shared memory
  - SMs → L2 Cache
- Time loop inside the kernel





Pershin I. et al. GPU Implementation of a Stencil Code with More Than 90% of the Peak Theoretical Performance // RuSCDays2019

SMs synchronization approaches

Asynchronous

kernel<<<blocks,threads>>>(...)

- Well-known standard CUDA approach
- Automatic sync at every stencil step
- Global memory is intensely used
- **Not suitable for RDD implementation**

Synchronous

cudaLaunchCooperativeKernel(kernel, blocks , threads ,...)

Cooperative Groups

- Official API (CUDA $\geq$ 6.1)
- All-SM barrier sync
- Generic instructions for any task
- RDD is fast enough

Semaphores

- Manual implementation
- SM-pairwise sync
- Task- and algorithm-specific code
- **RDD is even faster**

How to compare sync methods? Tau-model

Model parameters:

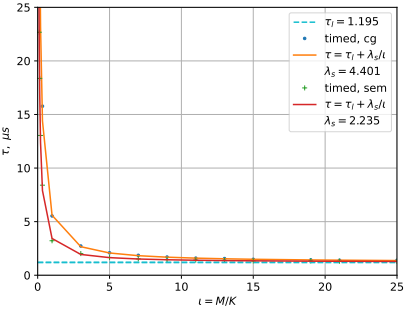
- N update iter-s
- each iter = M computing op-s and K data sync op-s
- elapsed $T(M, N, K)$ [sec]

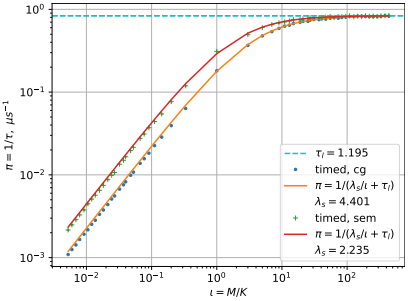
Model derivatives:

- *iteration time* $\tau(M, K) = \lim\limits_{N \to \infty} \frac{T(N,M,K)}{NM}$
- *computation time* $\tau_l = \lim\limits_{\substack{M \to \infty \\ K = const}} \tau(M, K)$
- *synchronization latency* $\lambda_s = \lim\limits_{\substack{K \to \infty \\ M = const}} [\tau(M, K + 1) - \tau(M, K)]$
- *arithmetic intensity* $\iota = \frac{M}{K}$
- *performance* $\pi = \frac{1}{\tau}$

Tau-model testing and fitting

- Model **quantifies** latency λ_s for CG and Sem syncs
- $\tau(\iota)$ **fits** well to $\tau = \tau_l + \frac{\lambda_s}{\iota}$
- Empirically,
  - $\tau_s = \frac{\lambda_s}{\iota}$ Little's law
  - $\tau = \tau_s + \tau_l$ Additive law
  - $\frac{1}{\pi} = \frac{1}{\pi_s} + \frac{1}{\pi_l}$ Harmonic law

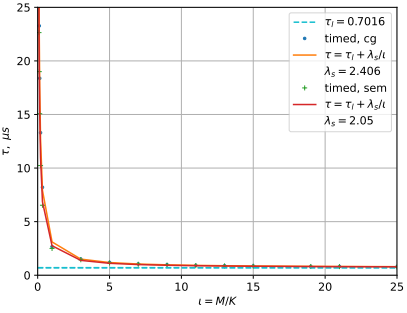


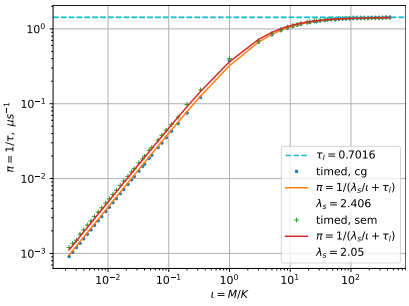


RTX 2070

Tau-model testing and fitting

- Model **quantifies** latency λ_s for CG and Sem syncs
- $\tau(\iota)$ **fits** well to $\tau = \tau_l + \frac{\lambda_s}{\iota}$
- Empirically,
  - $\tau_s = \frac{\lambda_s}{\iota}$ Little's law
  - $\tau = \tau_s + \tau_l$ Additive law
  - $\frac{1}{\pi} = \frac{1}{\pi_s} + \frac{1}{\pi_l}$ Harmonic law





RTX 3090

Conclusions

1. RDD for D3Q19 LBM is developed with two sync options available
2. New *tau-model* is developed, linking *overall performance* with *comp performance*, *sync latency*, and *arithmetic intensity*

$$\frac{1}{\pi} = \frac{1}{\pi_I} + \frac{\iota}{\lambda_s}$$

3. Found out
   - RTX 20: Semaphores are 2.0x faster than CG barriers
   - RTX 30: Semaphores are 1.2x faster than CG barriers

Thank you for your attention!

Contact: pershin2010@gmail.com