Lattice QCD (part 4)

Eloy Romero Alcalde

Jeffeson Lab

International School of Hadron Femtography
Sep 23 2024

Big pic

- We approximate a bunch of $\langle \mathcal{O}_i \rangle = \int \mathcal{O}_i(\boldsymbol{x}) e^{-S(\boldsymbol{x})} d\boldsymbol{x}$ as
- $\frac{1}{n} \sum_k \mathcal{O}_i(\boldsymbol{x}^{(k)})$ with $p(\boldsymbol{x}^{(k)}) \propto e^{-S(\boldsymbol{x}^{(k)})}$, drawing the samples with HMC
- $\boldsymbol{x}$ is a 4D lattice links field of random SU(3) variables

- Solving linear systems $D_{\boldsymbol{x}} \mathbf{u} = \mathbf{v}$ is the dominant computation, where $D_{\boldsymbol{x}}$ is the Dirac operator
- The linear systems are solved iteratively and accelerated with a multigrid

As the lattice spacing a and the quark mass m get smaller:

- The autocorrelation of HMC samples increases
- The linear systems become more ill-conditioned
- Each iteration takes longer

Multilevel integration

- We want to reduce the #samples in approx. $\langle \mathcal{O}_i \rangle = \int \mathcal{O}_i(\boldsymbol{x}) e^{-S(\boldsymbol{x})} d\boldsymbol{x}$
- We propose to compute instead $\langle \tilde{\mathcal{O}}_i \rangle = \int \tilde{\mathcal{O}}_i(\boldsymbol{x}) e^{-S(\boldsymbol{x})} d\boldsymbol{x}$, and
- do $\langle \mathcal{O}(\boldsymbol{x}) \rangle = \left\langle \tilde{\mathcal{O}}(\boldsymbol{x}) \right\rangle_{\boldsymbol{x} \in \text{level } 2} + \left\langle \mathcal{O}(\boldsymbol{x}) - \tilde{\mathcal{O}}(\boldsymbol{x}) \right\rangle_{\boldsymbol{x} \in \text{level } 1}$,
- where $\text{Corr}[\tilde{\mathcal{O}}, \mathcal{O}] \approx 1$, and
- the computation cost of sampling $\tilde{\mathcal{O}}$ is much cheaper than $\mathcal{O}$
- $\tilde{\mathcal{O}}(\boldsymbol{x}) \approx \mathcal{O}(\boldsymbol{x})$ but solving the linear systems less accurately

Solution of linear systems

Solving a linear system, $A\boldsymbol{x} = \mathbf{y}$, with A being a square matrix of size n:

- with Direct methods: cost $O(n^3)$, but machine precision
  Eg, LU based:
  1. Decompose $A = LU$, where L and U upper and lower triangular matrices
  2. $A^{-1}\mathbf{y} = U^{-1}(L^{-1}b)$, using forward and backward substitution alg.

- Iterative method: cost $O(\text{nonzeros}(A) \times k)$, for k iterations, variable precision
  Eg, Minimum Residual (only for positive definite A):
  1. $\mathbf{r} \leftarrow \mathbf{b} - A\boldsymbol{x}$, compute initial residual
  2. $\mathbf{p} \leftarrow A\mathbf{r}$, compute new search direction
  3. $\alpha \leftarrow (\mathbf{p}^\dagger\mathbf{r})/(\mathbf{p}^\dagger\mathbf{p})$
  4. $\boldsymbol{x} \leftarrow \boldsymbol{x} + \alpha\mathbf{r}$
  5. $\mathbf{r} \leftarrow \mathbf{r} - \alpha\mathbf{p}$
  6. Go to 2 if $\|\mathbf{r}\|$ is not small enough

Solution of linear systems

Solving a linear system, $A\boldsymbol{x} = \mathbf{y}$, with A being a square matrix of size n:

- with Direct methods:
  - Expensive for whole matrices
  - Difficult to parallelize
  - May play a role as preconditioners, (eg block jacobi)
- Iterative method:
  - Cheap if they converge fast
  - Preconditioners can be use to accelerate the convergence

Solution of linear systems

Consider the linear system of equations: $A\hat{\mathbf{x}} = \mathbf{b}$:

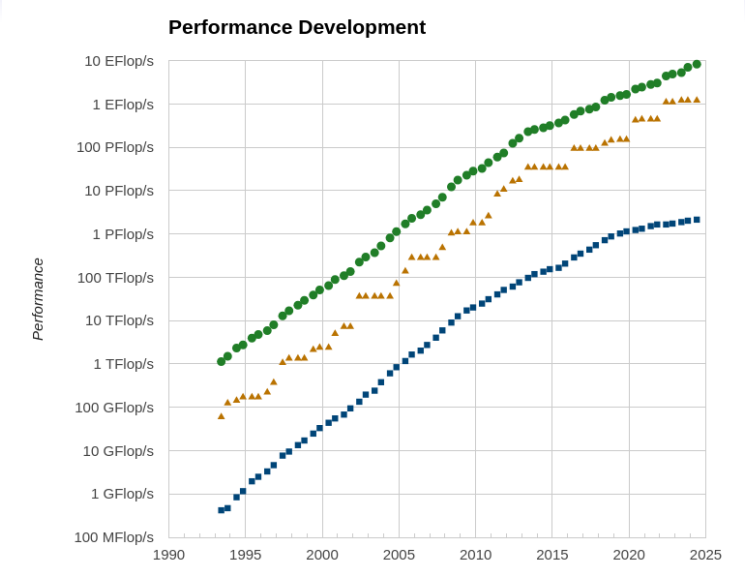
- All iterative methods are based on creating a polynomial p such that $p(A)\mathbf{b} \approx A^{-1}\mathbf{b}$
- Basic method
  1. Compute a basis of $V = [\mathbf{b},\ A\mathbf{b},\ A^2\mathbf{b}, \cdots, A^k\mathbf{b}]$
  2. Compute $A\boldsymbol{x} - \mathbf{b} \perp \text{span}\{V\}$ with $\boldsymbol{x} \in \text{span}\{V\}$, that is, $\boldsymbol{x} = V(V^*AV)^{-1}V^*\mathbf{b}$
  3. Stop if $\|A\boldsymbol{x} - \mathbf{b}\|_2$ is small enough
  4. Otherwise, set $\mathbf{b} \leftarrow \mathbf{b} - A\boldsymbol{x}$ and go to step 1

Recommended bibliography: Iterative Methods for Sparse Linear Systems, Y. Saad (online & free)

"The development of computational strategies in lattice QCD requires physical insight to be combined with an understanding of modern numerical mathematics and of the capabilities of massively parallel computers. When a new method is proposed, it should ideally be accompanied by a theoretical analysis that explains why it is expected to work out. However, in view of the complexity of the matter, some experimenting is often required. The field thus retains a certain empirical character."

Summer School on Modern perspectives in lattice QCD
Les Houches, August 3–28, 2009
Martin Lüscher

Evolution of computers



World fastest computers:

- top 1 – orange
- top 500 – blue
- sum 1-500 – green

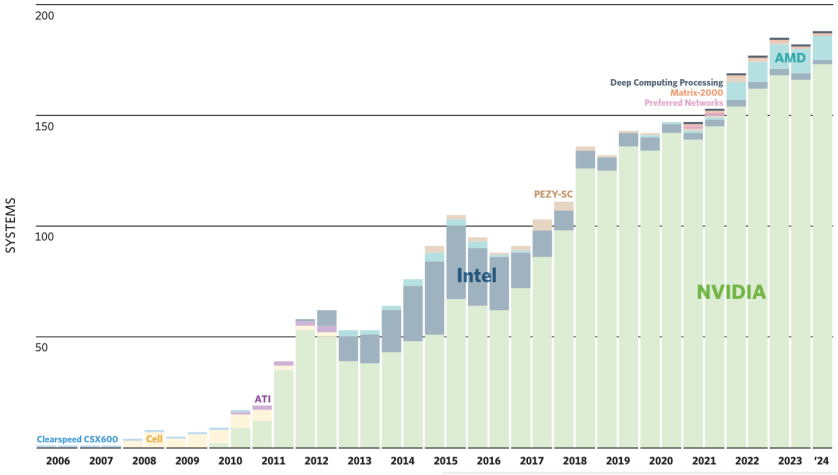
1 MFlop 10^6 flops
1 GFlop 10^9 flops
1 PFlop 10^{12} flops
1 TFlop 10^{15} flops
1 EFlop 10^{18} flops

Almost exp. growth!

source: top500.org

Evolution of computers

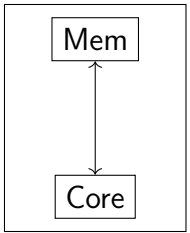
Number of systems with accelerators:



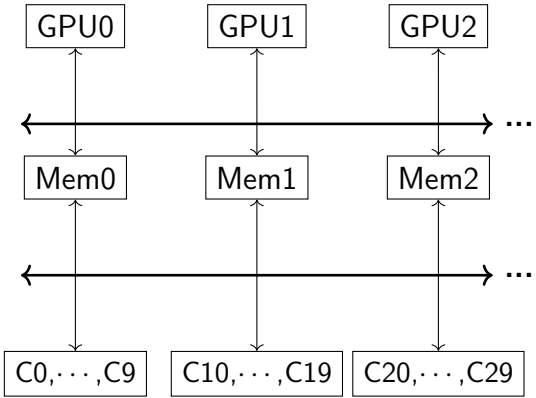
source: top500.org

Evolution of computers

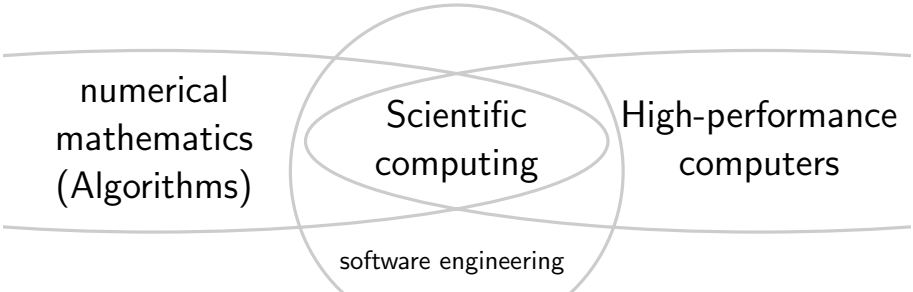
A node late 90s



A node today



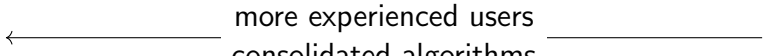
Scientific computing



- Mission: to make high-performance computers easy to use
- We are failing... up to some extend

Frameworks

Low-level (potentially very efficient)	Low-level frameworks (efficient)	High-level frameworks (somehow efficient)
C/C++, Fortran GPU kernels Vendor libs for BLAS (cu/rocBLAS, cu/rocSPARSE) Communication libraries (MPI, NCCL/RCCL)	PETSc, Trilinos	PyTorch, JAX Julia, Matlab/Octave



No solution fits all cases!

Frameworks

They offer:

- Parallelism and GPU support for distributing sparse and dense vectors and matrices
- Dense/sparse matrix-dense/sparse matrix multiplication
- Linear system solvers, eigenvalue solvers

They miss:

- Independent tensor view of the underneath layout
- Tensor contraction
- Efficient support for tensors, specially with small-size dimensions

General tips

- "Premature optimization is the root of all evil" – Donald Knuth
- Optimize code sections that amount a significant fraction of the total execution time
- Keep around simple versions of the code for checking correctness and benchmarking
- Writing low-level CPU/GPU kernels should be the last resort
- If it isn't tested, it doesn't work
- Thorough testing is challenging thou

Exercises

`https://github.com/eromero-vlc/summer-school-femto`