The computational challenges of Lattice Quantum Chromodynamics

C. Bonati

Dipartimento di Fisica & INFN, Pisa $\rightarrow$ Firenze (Italy)

Perspectives of GPU computing in Science

Roma, 26-28 September 2016

Outline

Quantum Chromodynamics (QCD)

From the Oxford dictionary point of view:

A quantum field theory in which the strong interaction is described in terms of an interaction between quarks mediated by gluons, both quarks and gluons being assigned a quantum number called "colour".

From the theoretical physics point of view:

$$\mathcal{L}_{QCD} = -\frac{1}{4} G^a_{\mu\nu} G^{a\,\mu\nu} + \sum_f \bar{\psi}^f (i\gamma^\mu D_\mu - m^f)\psi^f$$

with $f \in \{u, d, s, c, b, t\}$. Parameters: the dimensionless coupling g and the masses m_f of the six quark flavours.

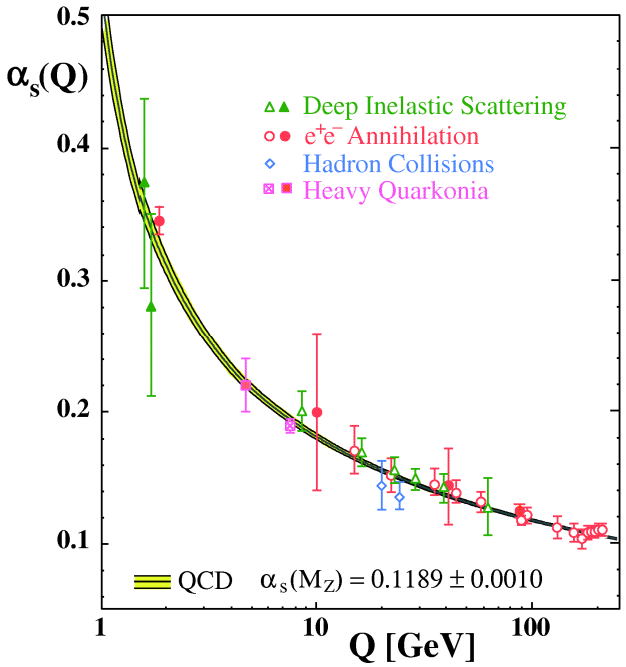
Quantum Chromodynamics (QCD)

Naive point of view

We know the Lagrangian of QCD: we know everything about QCD.

In practice the standard method used in theoretical high-energy physics to obtain quantitative predictions is perturbation theory:

- in QED the coupling at low energy is $\alpha_{em} \equiv e^2/(4\pi) \simeq 1/137$
- in QCD the coupling at low energy is $\alpha_s \equiv g^2/(4\pi) \simeq 1$



Asymptotic freedom / Infrared slavery

- If $E \gg 1\,\mathrm{GeV}$ the theory is perturbative and usual methods work.
- If $E \lesssim 1\,\mathrm{GeV}$ the theory is nonperturbative.

from hep-ex/0606035

Why bothering of low energy QCD?

- Free quarks have never been directly observed in nature. Quarks are confined into hadrons and the typical scale of light hadrons is about $1\,\mathrm{GeV}$. If we want to connect the fundamental theory with the real world we need low-energy QCD.
  - does QCD Lagrangian imply quark confinement?
  - can we predict the masses of hadrons?
  - ab-initio nuclear physics?

- What about thermodynamics? At $T = 0$ we experimentally observe confinement, at high energy the coupling gets smaller and smaller: the existence of an high temperature deconfined phase seems reasonable.
  - does a high temperature deconfined phase actually exist?
  - does a phase transition exist at finite temperature?
  - can we predict the thermodynamical properties of strongly interacting matter?

Lattice QCD (LQCD)

Quantum Field Theories (QFT) and Statistical Physics (SP) are much more similar than they look at first sight:

$$\text{QFT} \longleftrightarrow e^{i\int \mathcal{L}\mathrm{d}^3x\mathrm{d}t} \qquad \text{Feynman path integral}$$

$$\text{SP} \longleftrightarrow e^{-E/T} \qquad \text{Boltzmann distribution}$$

If we use $t = i\tau$ (i.e. we use an imaginary time, Wick rotation), then QFT becomes formally identical to SP, modulo the identification

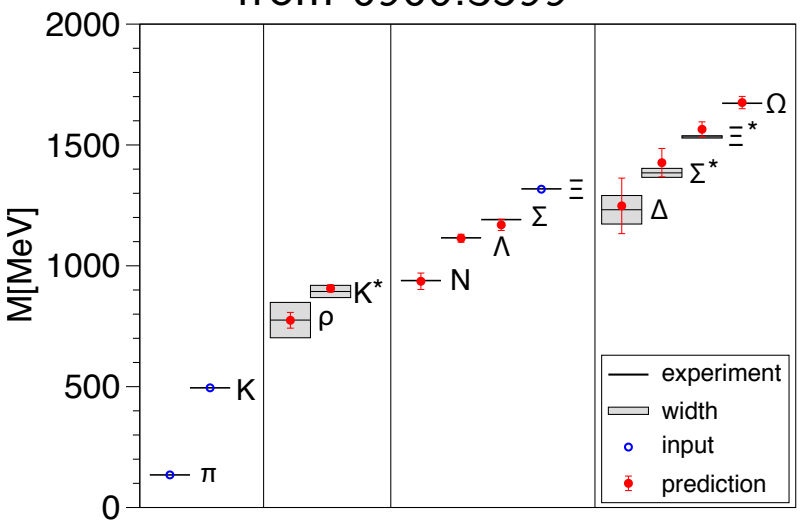
$$\int \mathcal{L}_E \mathrm{d}^3x\mathrm{d}\tau \longleftrightarrow E/T \qquad (\text{with } \mathcal{L}_E(x,\tau) = \mathcal{L}(x,t))$$

The statistical physics side of QFT: the values of the field at every point of the space-time are random variables with distribution $\exp(-\int \mathcal{L}_E \mathrm{d}^3x\mathrm{d}\tau)$.

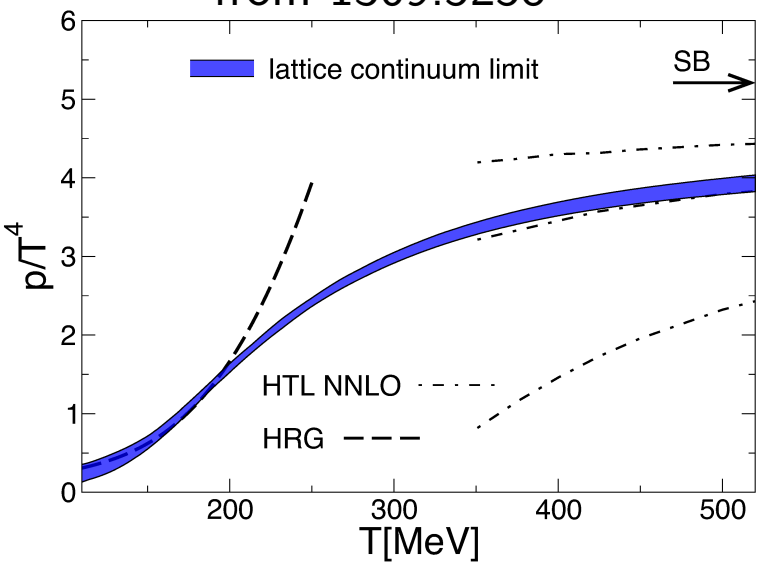
To use standard Monte Carlo methods we need a finite dimensional distribution so we use a space-time discretization: a lattice.

LQCD: does it work?

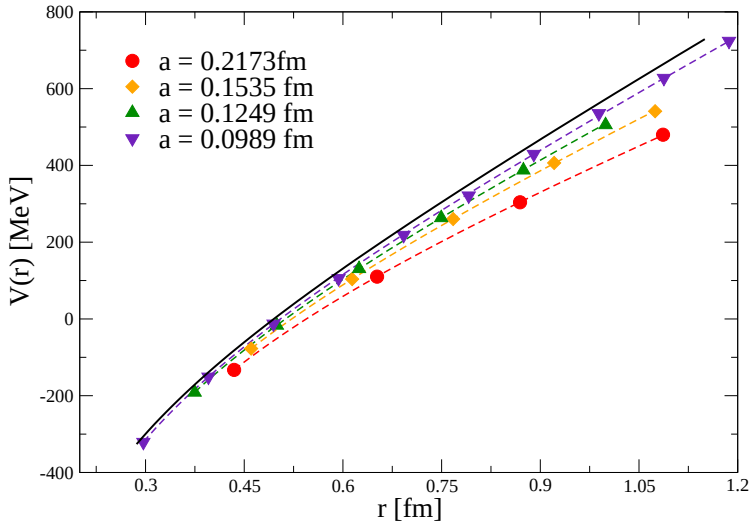
from 0906.3599



from 1309.5258



from 1607.08160



It works very well for a lot of stuff!
Notable exceptions:

- QCD at finite density
- kinetic coefficients (viscosity, conductibility,. . .)

LQCD: why is it difficult?

Scale separation

The "typical scale" of nonperturbative QCD is $\Lambda_{QCD} \approx 1\,\mathrm{GeV}$, the masses of the first four quark flavours are $m_u \approx 0.002\mathrm{GeV}$, $m_d \approx 0.005\,\mathrm{GeV}$, $m_s \approx 0.1\,\mathrm{GeV}$, $m_c \approx 1.3\,\mathrm{GeV}$.
To obtain reliable results we must have control on energies that span around four order of magnitudes.
(problem analogous to the one of ab-initio quantum chemistry)

The interaction is very complicated

The distribution we have to sample in the Monte Carlo is of the form $P(A) \sim \det[M(A)]e^{-S_G(A)}$, where $S_G(A)$ is an "easy" (almost) next neighbourhood interaction and $M(A)$ is a large, sparse and structured matrix.

LQCD Monte Carlo algorithm

We have to sample the Boltzmann distribution with energy

$$E \sim S_G(A) + \log[\det M(A)] \sim S_G(A) + \phi^\dagger \frac{1}{M(A)^\dagger M(A)} \phi$$

The standard algorithms of statistical mechanics are:

Metropolis algoritmh change randomly the configuration, compute ΔE, accept the new configuration with probability $\exp(-\Delta E/T)$.

Molecular/Langevin Dinamics integrate numerically the equations of motion (eventually with stochastic noise).

Drawbacks of these algorithms:

- for the acceptance probability to be reasonable, only small local updates have to be performed, but the energy is very non-local and the computation of ΔE scales badly with volume ($\sim V^3$).
- M(/L)D is computationally cheaper since the update is global, but it is not stochastically exact: we need the limit $\Delta t \to 0$.

LQCD Monte Carlo algorithm

Hybrid Monte Carlo (HMC): we first of all add fictitious momenta that enter quadratically in the energy of our system, then (starting from x_i)

- generate the initial momenta p_i with gaussian distribution
- compute the initial energy E_i
- evolve the variables by numerically solving the EoS in the fictitious time (and arrive to x_f, p_f)
- compute the final energy E_f
- accept the update $(x_i, p_i) \rightarrow (x_f, p_f)$ with probability $\exp(-\Delta E/T)$

Such an algorithm takes the best of Metropolis and MD: it is a global stochastically exact algorithm.
If the integration step is too coarse, the acceptance probability degrades, but the algorithms always converges to the correct result.

HMC: the hard step

To compute the force in the MD step of HMC one has to solve

$$M(A)x = b \qquad \text{(or variants of it)}$$

where b is a random vector. Some general properties of $M(A)$ are:

- it is a large sparse matrix, typically of order $10^6 \times 10^6$
  (the order is directly proportional to the physical volume)
- it is a structured matrix, the basic blocks being $SU(3)$ matrices
  (because there are 3 color charges)
- it is ill conditioned ($\kappa \gtrsim 10^5$)
  (because $m_{u,d} \ll \Lambda_{QCD}$ and chiral symmetry breaking)

This equation has to be solved $\mathcal{O}(10)$ times for a single HMC step, and $\mathcal{O}(10^5 \div 10^6)$ HMC steps can be necessary.

The CG and its friends

The most used methods to solve the equation $Mx = b$ are Krylov solvers. The prototypical example is the Conjugate Gradient method:

x_0 arbitrary starting point;
$r_0 = p_0 = b - Mx_0$;
while $\|r_n\| > \epsilon$ **do**

$$\beta_n = -\|r_n\|^2 / (p_n, Mp_n)$$
$$x_{n+1} = x_n - \beta_n p_n$$
$$r_{n+1} = r_n + \beta_n M p_n$$
$$\alpha_{n+1} = \|r_{n+1}\|^2 / \|r_n\|^2$$
$$p_{n+1} = r_{n+1} + \alpha_{n+1} p_n$$

end

The fundamental operation that need to be optimized to achieve high performance is the matrix $\cdot$ vector operation.

LQCD on GPU

Since the building blocks of LQCD are linear algebra operations, GPU appears as perfect candidates for LQCD simulations.

The seminal work that used GPU for LQCD purposes was

G. I. Egri, Z. Fodor, C. Hoelbling, S. D. Katz, D. Nogradi and K. K. Szabo,
Lattice QCD as a video game
Comput. Phys. Commun. **177**, 631 (2007) [hep-lat/0611022].

that appeared the 21^{st} November 2006, in which OpenGL (GL not CL!) was used. The 23^{rd} June 2007 the first CUDA release appeared and the first LQCD application of CUDA was

K. Barros, R. Babich, R. Brower, M. A. Clark and C. Rebbi,
Blasting through lattice calculations using CUDA
PoS LATTICE **2008**, 045 (2008) [arXiv:0810.5365 [hep-lat]].

(Babich and Clark now work at Nvidia)

The main obstacle: bandwidth

The elementary blocks of the $M(A)$ matrix are 3×3 complex matrices. If we consider a single precision computation, the product

$$(3 \times 3 \text{ complex matrix}) \cdot (\text{complex } 3-\text{vector})$$

requires 72 floating number operations and 96 bytes of memory transfer. This operation (and thus the whole algorithm) has low computational intensity and the algorithm is strongly bandwidth limited.

E.g. on Nvidia K80 the maximum bandwidth is 480 GB/s and the peak performance in single precision is 8.7Tflops. The maximum expected performance is around 0.4Tflops, i.e. around 5% of the peak.

Double precision relative effectiveness is slightly better (the effective bandwidth is halved but the peak performance is reduced by more than a factor of two, being 2.9Tflops) but in absolute terms it is a factor 2 slower than the single precision.

Partial workarounds

- We know that the 3×3 complex matrix is in fact an $SU(3)$ matrix, thus the third row can be computed on fly as the vector product of the first two rows. In this way we gain a factor $4/3$ in efficiency. In principle one could do even better, since only 8 real numbers are required to uniquely identify an $SU(3)$ matrix, however the reconstruction algorithm is not stable enough in this case.
- Use mixed precision Krylov solvers (a.k.a defect-correction or reliable updates algorithms). The idea is that Krylov algorithms are very stable, so much that it is possible, with minor changes, to obtain an high precision result using almost always low precision operations. It is thus possible to use

$$\text{half precision} \rightarrow \text{single precision} \rightarrow \text{double precision}$$

Warning: mixed precision cannot be used in case we need to solve $(M + \sigma_i)x = b$ for several σ_i since shifted Krilov solvers cannot be restarted.

Memory layout

The "natural" memory layout for a LQCD code would be an Array of Structure (AoS), with "structure"$=SU(3)$ matrix, however to achieve high performance it is important to use a Structure of Array (SoA) memory layout.

On old GPU architectures SoA was absolutely fundamental to have coalesced memory accesses (and also textures helped). More recently this constraint has been largely reduced, however it is still strongly suggested to prefer SoA to AoS.

Possible bonus: on old CPUs there was no real gain in using SoA, and in fact it was sometimes worst than AoS. Massive vectorization in CPUs (e.g. Intel KNL) is changing this: the strategies suggested by CPU vendors to write efficent code by now resemble very much the ones for GPU.

Single code for heterogeneous architectures?

OpenCL First proposal for an unified language for CPUs and GPUs. It is structurally very similar to CUDA and apparently Nvidia does not like it very much...

OpenACC Programming standard that uses compiler directives, it is basically a multi-architecture version of OpenMP.

Advertising: "Initial OpenACC implementation required only minor effort, and more importantly, no modifications of our existing CPU implementation". In red Nvidia emphasis, in blue my emphasis.

In the Pisa/Ferrara collaboration we now have a complete working MPI/OpenACC LQCD code and we are going to

- compare its performance w.r.t. the previous CUDA version of the code (preliminary results indicate a loss of performance around 20%)
- compare the performances on different architectures: Nvidia GPU, ATI GPU, Broadwell CPU, Knights Landing, ...

Conclusions

- Lattice QCD is at present the only method to extract systematically improvable first-principle results concerning strong interactions in the non-perturbative regime.
- To perform reliable LQCD simulations requires a tremendous amount of computing power.
- In the LQCD community there is a strict connection between the development of new physical insights, the introduction of new algorithms, the building of new machines and the efficient implementations of the algorithms on the target machines.
- It will be interesting to see how different accelerator technologies that now are competing with each other will likely mix together in the future.

Thank you for your attention!