Common code system for the lattice QCD simulations

Shinji MOTOKI (KEK, A04 team)

for

Bridge++ Code development Project

H. Matsufuru gives a talk instead, with a little update
MEMBERS


Also advised by H. Tadano, A. Imakura, Seminars by M. Sato and A. Nakamura

Programmers, reviewers and users are wanted. Any interested people are welcome anytime!
Grant-in-Aid for Scientific Research on Innovative Areas
“Research on the Emergence of Hierarchical Structure of Matter by Bridging Particle, Nuclear and Astrophysics in Computational Science”
The A04 team
“Interdisciplinary algorithms and computer simulations”
http://bridge.kek.jp/A04/ (H. Matsufuru's talk)

HPCI Strategic Program Field 5
“The origin of matter and the universe”
http://www.jicfus.jp

– So far 60 meetings held every 1-2 weeks
– Advices given by experts in computer science and applied mathematics
● **Portability:** running on various environment, from notebook PC to supercomputers

● **High performance:** fully making use of wide range of architecture, with state of the art techniques

● **Easy to understand** even for beginners

● **Easy to extend** to test new ideas
WHY COMMON CODE?

- **Research environment may changes**
  Collaboration members frequently come and go.
  - who maintains the code?
  - communication problem might occur.
  Machine architecture may change.
  - have to rewrite a new code for updated machines?

- **Demands for “standard”**
  Users should concentrate on their physics projects.
  Generated data can be shared by different groups.
  Common language on the calculations is convenient.

- **Why not using existing codes?**
  E.g. Chroma and CPS++ are widely used in the community.
  We want a code completely under control of ourselves from foundation
  - Quick response to user’s requests
  --Detailed documentation and consulting service
  - Accumulating experiences of the development is important. to keep technology
Our aim:
well-organized portable code with a good performance, allowing beginners to carry out “professional simulations”

C++ language:
Design by the object oriented programming
Stick to the standard libraries for portability
Parallelized with MPI

Documentation
Doxygen is helpful: comments embedded in the code
Detailed manual in English/Japanese

Covering all basic calculations in Lattice QCD:
Gauge configuration generation + measurements
Commonly used lattice fermions
ILDG data format
Maximum flexibility in simulation parameters
WHAT HAVE BEEN IMPLEMENTED

Ver.1.0 public release 24 July 2012

- **Gauge action:** Plquette, Rectangular
- **Fermion action:** Wilson, Clover, staggered, overlap, domain-wall
- **Link smearing:** APE or HYP x stout (+ projection)
- **Linear solvers:** CG, BiCGStab, GMRES, etc. + shift solver (CG)
- **Eigen solvers:** Implicitly restarted Lanczos (for Hermitan matrix)
- **HMC:** multi-time step, Hasenbusch, Omelyan integrator, Rational HMC
- **Gauge fixing:** (Coulomb, Landau)
- Schrodinger functional boundaries, isospin chemical potential
- meson/baryon correlators (Dirac/chiral spinor representations)
- Wilson loop, Polyakov loop, etc.

- **ILDG format is supported in configuration data I/O**

- **Now in progress:** multi-thread (openMP or pthread ?), GPU (OpenCL)
Example: solver and fermion operator

Class diagram: relation between classes

Solver does not distinguish which fermion operators:
Fopr (base class) defines interface (virtual method)

Virtual base class defines solver interface

Virtual base class defines fermion operator interface

Inherit class implements each practical algorithm

Inherit class implement each fermion operator
CONFIRMATION

- Example of comparison to literature
  Reference: BMW Colab. JHEP 1108(2011) 148
  $N_f=2+1$, 2 stout-HYP (2HEX), $16^3 \times 32$ lattice

BG/Q @KEK, 3% of peak performance, flat MPI for all the cores
**Trac/Subversion: joint development by the version control system**

**Subversion: version control system of code set**

**trac: project control system**

Organized information using the wiki

← screen shot (trac)

↓ repository browser
WEB SITE

- Source Code
- Release information
- Progress of development
- Manuals / User’s guide
- Confirmation reports etc.

Please access to
http://suchix.kek.jp/bridge/Lattice-code/
(Japanese only now)
- First-step guide, implementation note, etc.
- doxygen
ROAD MAP

● **Action/Algorithms to be implemented**
  - Now Wilson/clover fermions are available in public version
  - Staggered (standard), domain-wall, overlap fermions are almost ready

● **Improvement of design**
  - General gauge group, fermion representations

● **Performance tuning**
  - On Hitachi SR, about 5%
  - On IBM Blue Gene/Q, less than 5% --- being improved
  - Shared memory parallelization
  - Framework to use accelerators: next page
ACCELERATORS

- General framework to use various accelerators (GPGPU, Cell B.E., MIC, etc.)

  We employ **OpenCL** (implemented, now being tuned)

  -- *Open Computing Language* (OpenCL) is a framework for writing programs that execute across heterogeneous platforms consisting of central processing units (CPUs), graphics processing units (GPUs), DSPs and other processors.
New lattice QCD code Bridge++ has been developed aiming at:

- Developing research environment
  - to skip unnecessary efforts of coding while getting high performance
  - to remove barriers of communication between researches/beginners
  - to share experiences, ideas and data

- Making use of knowledge of different fields
  - applied mathematics (algorithms)
  - computer science, software design

One of the goals of this program

The project is still in the early stage.
We strongly need your suggestions, contributions, and feedbacks
Thank you for your attention.

http://suchix.kek.jp/bridge/Lattice-code/