



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

S. Aoki, T. Aoyama, G. Cossu, T. Doi, S. Hashimoto,
N. Ishii, K-I. Ishikawa, K. Kanaya, T. Kaneko,
Y. Kuramashi, H. Matsufuru, S. Motoki, Y. Namekawa,
H. Nemura, J. Noaki, K. Ogawa, H. Saito, S. Sasaki,
Y. Taniguchi, S. Ueda, N. Ukita, T. Yoshie.

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!**



SUPPORTED BY

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



DEVELOPMENT POLICY

- **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



PROFILE

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 Hermitian 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)

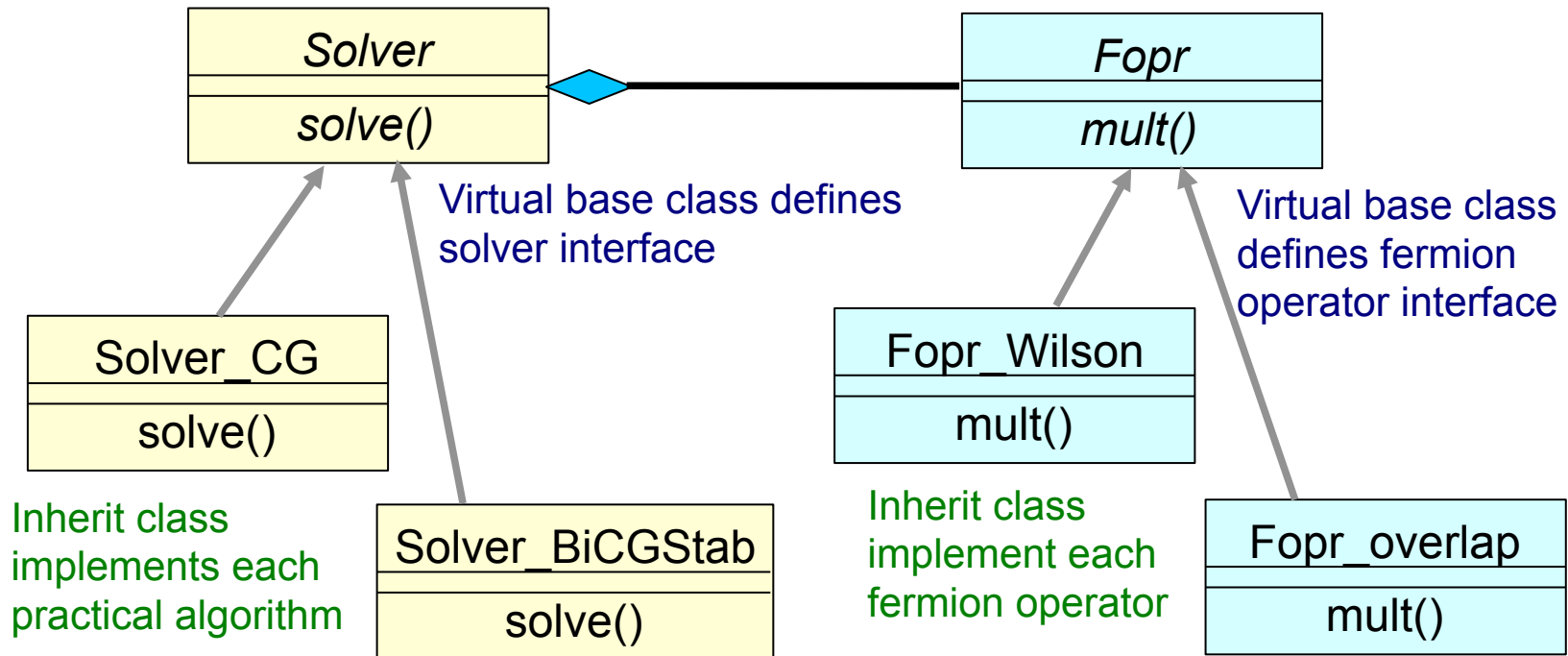


DESIGN

- Example: solver and fermion operator

Class diagram: relation between classes

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



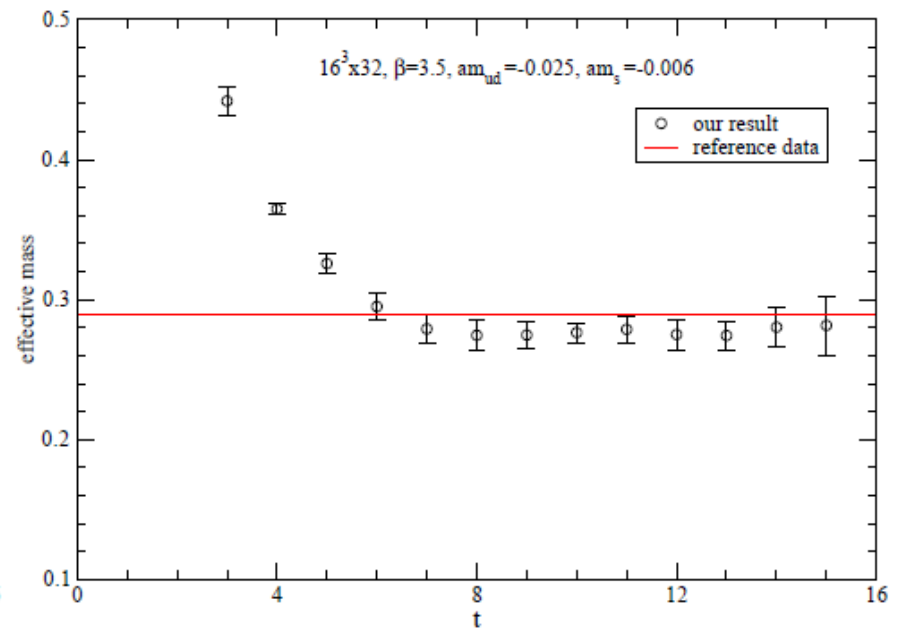
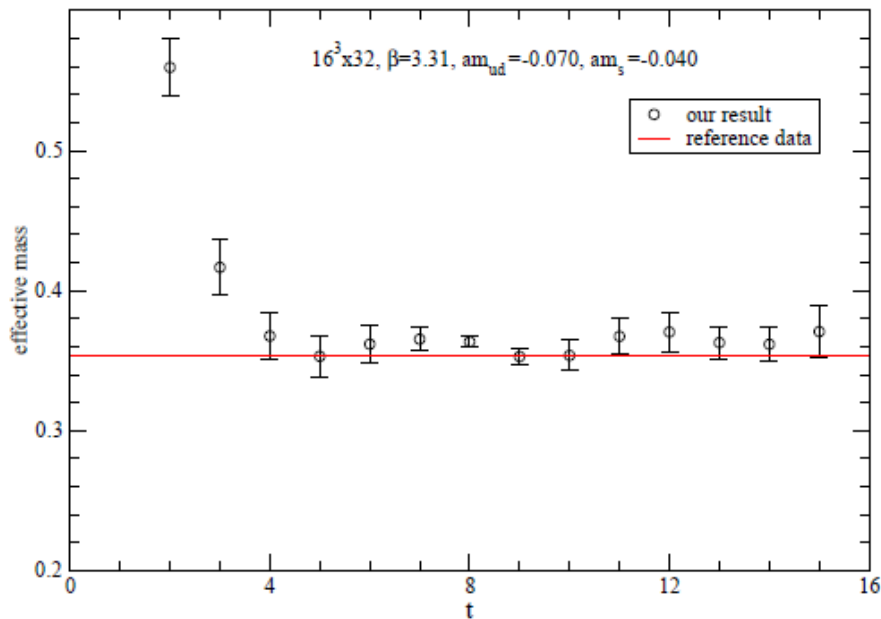


CONFIRMATION

- 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



DEVELOPMENT

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

code1.jldg.org/trac/bridge/wiki

smotoki としてログイン中 | ログアウト | ユーザ設定 | ヘルプ/ガイド | Trac について

Wiki | タイムライン | ロードマップ | リポジトリブラウザ | チケットを見る | チケット登録 | 検索 | 管理

スタートページ | ページ一覧 | ページ履歴 | 最終更新

lattice QCD 共通コードプロジェクト Bridge++

新領域研究「素粒子宇宙論による計算基礎科学に基づいた重層的物質構造の解明」および 計算基礎科学連携拠点 次世代スーパーコンピュータ戦略プログラム分野 5「物質と宇宙の起源と構造」（領域代表：青木慎也）では、QCD (Quantum Chromodynamics) を含む格子ゲージ理論のシミュレーションのためのコードを開発しています。

様々な格子作用やアルゴリズムを適用可能で、ノートPCから超並列計算機まで幅広いアーキテクチャに対応し、最先端の研究に必要なパフォーマンスを実現でき、なおかつ使い易いものを目指しています。2009年10月15日筑波大で1回目のフリーディスカッションを行って以来、勉強会などを重ねながら、設計、開発を続けています。興味のある方はお気軽にご参加下さい。また、開発に参加していただける方は経験問わず歓迎します。

管理人 上田 恒(sueda_AT_post.kek.jp)、元木 伸治(smotoki_AT_post.kek.jp)

Bridge++β コードセットの使い方

SVNユーザー登録

β版開発にはSVNユーザー登録が必要です。こちらのUserIDをお持ちの方はbridge++SVNに既にアクセス可能です。新規登録が必要な方は管理人までご連絡ください。

※TracのログインIDとは異なります。Tracのユーザー登録については、#Tracのユーザー登録を参照してください。

Tracのユーザー登録

TracのユーザーIDの登録はご自分で行えます。下記（アカウントの作成）に沿って登録してください。登録をしなくても閲覧は可能ですが、登録しログインする事によって、Tracのwiki等が編集可能になります。また、ユーザー作成時にメールアドレスを登録すると、自分に関するバグ報告などがメールで送信されるようになります。

- アカウントの作成

コードの入手先

- subversion: <http://code1.jldg.org/repos/bridge>

← screen shot (trac)

↓ repository browser

code1.jldg.org/trac/bridge/browser

smotoki としてログイン中 | ログアウト | ユーザ設定 | ヘルプ/ガイド | Trac について

Wiki | タイムライン | ロードマップ | リポジトリブラウザ | チケットを見る | チケット登録 | 検索 | 管理

最終更新 | リビジョンログ

root

移動する: [] リビジョンを見る: []

ファイル名	Size	Rev	Age	最終更新
branches		804	2 日	sueda: make pointer const.
tags		423	5 か月	sueda: tag422_120710 form trunk@422
trunk		795	13 日	namekawa: merge 792:794, including YAML to Tests (part 1)

属性 svnignore の設定値
trunk/build

Note: リポジトリブラウザについてのヘルプは [TracBrowser](#) を参照してください。

変更箇所を見る...

Powered by Trac 0.11.6.Ja1
by Edgewall Software.
Translated by インテグレート株式会社

Visit the Trac open source project at
<http://trac.edgewall.org/>



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)





DOCUMENTATION

- First-step guide, implementation note, etc.
- doxygen

Class list → function of each Classes

Bridge++ Ver. 1.0.3

Main Page Namespaces **Classes** Files Directories Search

Class List Class Index Class Hierarchy Class Members

▼ Bridge++
Lattice QCD common code development Project

▼ Class List

- Action
- Action_F_Ratio
- Action_F_Rational_frame
- Action_F_Rational_frame_SF
- Action_F_Standard
- Action_F_Standard_eo
- Action_F_Standard_lex
- Action_F_Standard_SF
- Action_G_Plq
- Action_G_Plq_SF
- Action_G_Rectangle
- Action_G_Rectangle_SF
- Communicator::Base
- Communicator::Impl::Base
- Bridge::BridgeIO
- Builder_Integrator
- Channel
- ChannelSet
- CommonParameters
- Communicator
- Communicator::Impl
- Corr2pt_4spinor
- Corr2pt_Wilson_SF

Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

Action	Base class of HMC action class family
Action_F_Ratio	HMC action for Hasenbusch preconditioned fermions
Action_F_Rational_frame	Action class for RHMC, with externally constructed Fopr_Rational
Action_F_Rational_frame_SF	Action class for RHMC, with externally constructed Fopr_Rational
Action_F_Standard	Standard fermion action for HMC
Action_F_Standard_eo	Standard even-odd preconditioned fermion action for HMC
Action_F_Standard_lex	Standard fermion action for HMC
Action_F_Standard_SF	Standard fermion action with SF BC for HMC
Action_G_Plq	HMC action class for plaquette gauge action
Action_G_Plq_SF	HMC action class for plaquette gauge action with SF BC
Action_G_Rectangle	HMC action class for rectangular gauge action
Action_G_Rectangle_SF	HMC action class for rectangular gauge action with the SF BC
Communicator::Base	
Communicator::Impl::Base	
Bridge::BridgeIO	
Builder_Integrator	Builder of MD integrator for HMC
Channel	
ChannelSet	
CommonParameters	Common parameter class: provides parameters as singleton
Communicator	Communication library which wraps MPI
Communicator::Impl	
Corr2pt_4spinor	Two-point correlator for Wilson-type fermions
Corr2pt_Wilson_SF	

Generated on Mon Aug 27 2012 14:12:12 for Bridge++ by [doxygen](#) 1.7.5.1

Bridge++ Ver. 1.0.3

Main Page Namespaces **Classes** Files Directories Search

Class List Class Index Class Hierarchy Class Members

Fopr_Chebyshev
Fopr_Clover
Fopr_Clover_eo
Fopr_Clover_imp
Fopr_Clover_SF
Fopr_Clover_SF_imp
Fopr_CloverTerm_eo
Fopr_CRS
Fopr_eo
Fopr_Rational
Fopr_Rational_SF
Fopr_Smeared

Fopr_Wilson Class Reference

Public Member Functions | Private Member Functions | Private Attributes

#include <fopr_wilson.h>

Inheritance diagram for Fopr_Wilson:

```
graph TD
    Fopr --> Fopr_Wilson
```

List of all members.

Public Member Functions

Fopr_Wilson()	
Fopr_Wilson(std::string repr)	
~Fopr_Wilson()	
void set_parameters(const Parameters_Fopr_Wilson &prms)	
void set_parameters(const double CKs)	
void set_parameters(const double CKs, const std::valarray<int> > bc)	
void set_config(Field *g)	
void setting_pointer_to_the_gauge_configuration.	
const Field mult(const Field &f)	multiplies fermion operator to a given field and returns the resultant field.
const Field mult_dag(const Field &f)	hermitian conjugate of mult(const Field &f).
void mult(Field &v, const Field &f)	multiplies fermion operator to a given field (2nd argument)
void mult_dag(Field &v, const Field &f)	hermitian conjugate of mult(Field &f, const Field &f).
void set_mode(std::string mode)	

Generated on Mon Aug 27 2012 14:12:12 for Bridge++ by [doxygen](#) 1.7.5.1



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

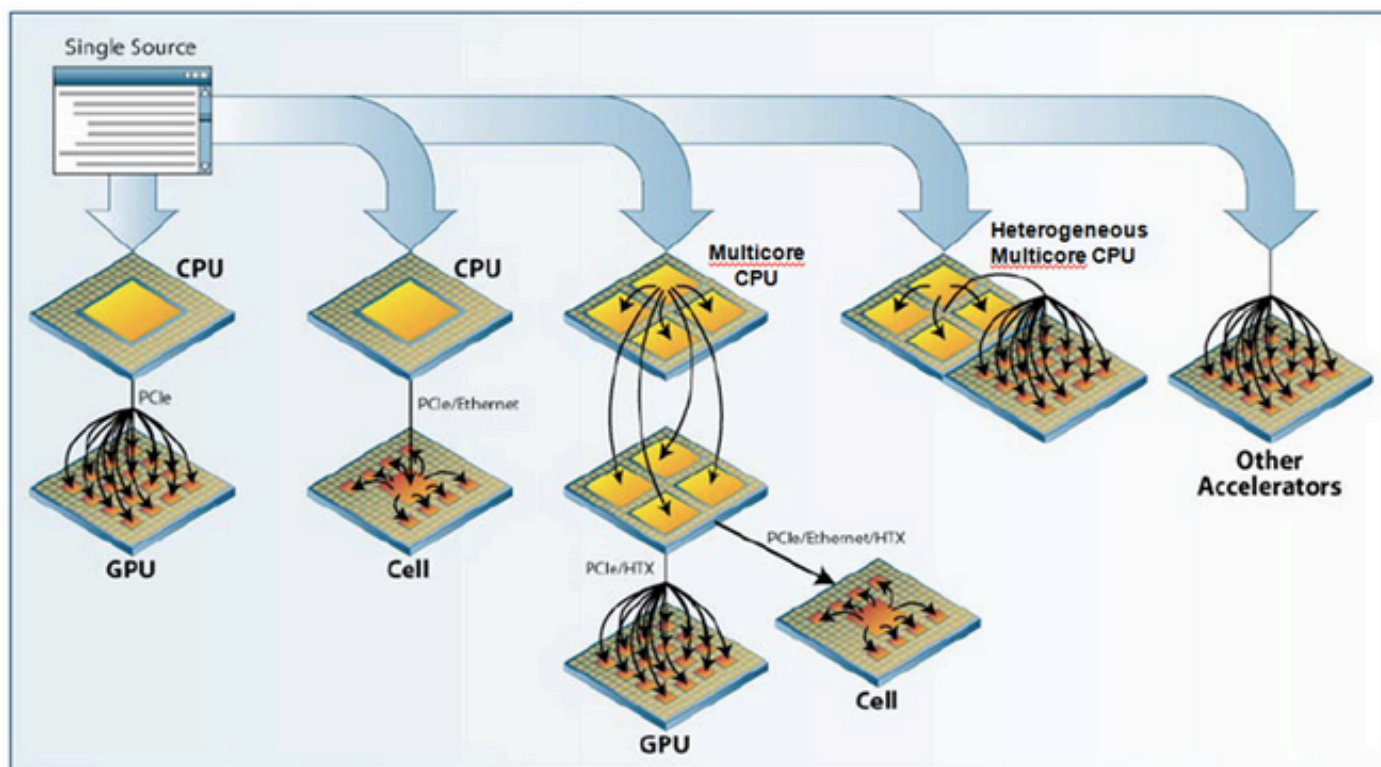


ACCELERATORS

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

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

- **Open** **C**omputing **L**anguage (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.





SUMMARY

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



DEMONSTRATION



Thank you for your attention.

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