QCD in GPU

Ting-Wai Chiu (趙挺偉)

Department of Physics
Center for Quantum Science and Engineering
National Taiwan University

GPU Technology Conference
Beijing, China
December 14-15, 2011

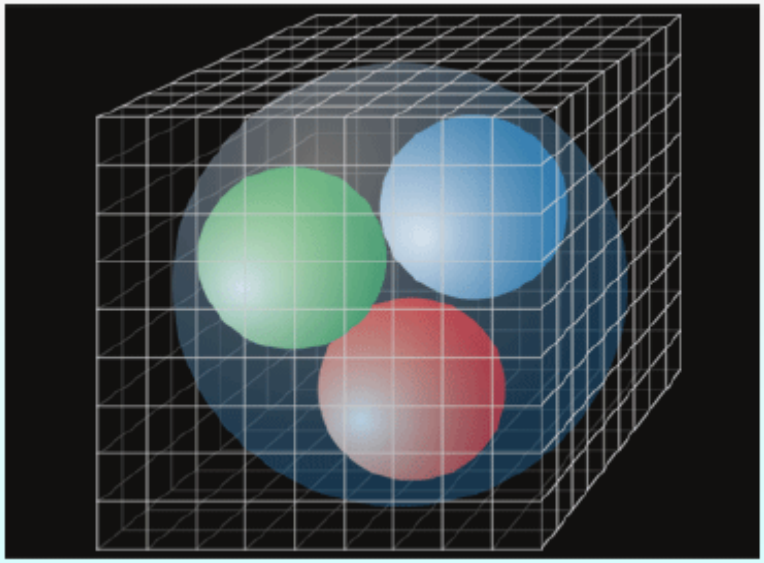
Quantum Chromodynamics (QCD)

- **The quantum field theory for the strong interaction between quarks and gluons** which build up the hadrons (e.g., neutron, proton, pion, etc.).

- QCD provides the framework to understand the nuclear force/energy from the first principles.

- QCD plays an important role in the evolution of the early universe, from the quark gluon "plasma" phase to the hadron phase.

◆ To solve QCD is a grand challenge among all sciences. The most promising approach to solve QCD nonperturbatively is to discretize the continuum space-time into a 4 dimensional lattice (lattice QCD), and to compute physical observables by Monte Carlo simulation.



Outline

- **Introduction**
- **Lattice Dirac Operator : Quark Matrix**
- **Cuda Kernels for Conjugate Gradient**
- **Conclusion & Outlook**

References:

Taiwan Lattice QCD Collaboration (TWQCD):
arXiv: 0911.5029, 1101.0405, 1101.0402, 1101.0423,
1105.4414, 1109.3675

Quarks

Quarks are spin $\frac{1}{2}$ Dirac fermions carrying color, and there are 6 species (flavors) of quarks.

u	c	t		u	c	t		u	c	t
d	s	b		d	s	b		d	s	b

Hadrons are color singlets composed of quarks

$P = uud$ + antisym. in color, Proton

$N = udd$ + antisym. in color, Neutron

$\pi^+ = \bar{d}u + \bar{d}u + \bar{d}u$, Pion

The nuclear force between nucleons emerges as residual interactions of QCD

Quark field

$\psi(x, y, z, t, c, \alpha)$ Grassman element

$c = 1, 2, 3$ Color index

$\alpha = 1, 2, 3, 4$ Dirac index

Quarks interact with each other by emitting/absorbing Gluons

Gluon field

There are eight kinds of Gluons.

The gluon field can be represented by a

3×3 unitary matrix.

$$U^{\mu}(x,y,z,t) = \begin{pmatrix} U^{\mu}_{11} & U^{\mu}_{12} & U^{\mu}_{13} \\ U^{\mu}_{21} & U^{\mu}_{22} & U^{\mu}_{23} \\ U^{\mu}_{31} & U^{\mu}_{32} & U^{\mu}_{33} \end{pmatrix}, \quad \mu = \hat{x}, \hat{y}, \hat{z}, \hat{t}$$

$$U^{\mu\dagger}U^{\mu} = U^{\mu}U^{\mu\dagger} = I$$

Gluon fields on the Lattice

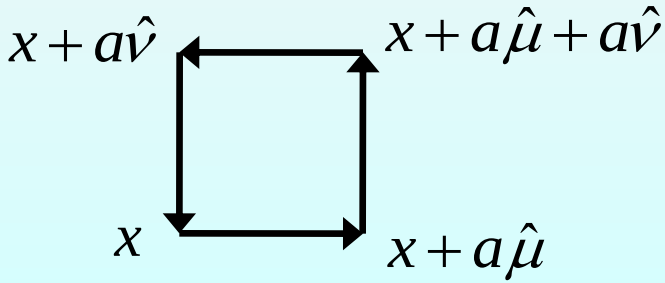
The $SU(3)$ color gluon field $A_\mu(x)$ are defined on each link connecting x and $x+a\hat{\mu}$, through the link variable

$$U_\mu(x) = \exp\left[iagA_\mu\left(x+\frac{a}{2}\hat{\mu} \right)\right]$$

Then the gluon action on the lattice can be written as

$$S_g[U] = \frac{6}{g^2}\sum_{\text{plaquette}}\left[1-\frac{1}{3}\text{Re } tr(U_p)\right] \xrightarrow{a\to 0} \int d^4x \frac{1}{2} tr\left[F_{\mu\nu}(x)F_{\mu\nu}(x)\right]$$

where $U_p = U_\mu(x)U_\nu(x+a\hat{\mu})U_\mu^\dagger(x+a\hat{\nu})U_\nu^\dagger(x)$



The Quark Matrix (Lattice Dirac Operator)

$$D = \left(D_{ij} \right)$$

$$i = (a, \alpha, x) \qquad j = (b, \beta, y)$$

$a, b = 1, 2, 3$ (color indices)

$\alpha, \beta = 1, 2, 3, 4$ (Dirac spinor indices)

$x, y = 1, \cdots, N_{sites}$ (Lattice site indices)

Salient Features of the Quark Matrix

- D is prohibitively large for exact solvers.
- In general, D is a sparse matrix, since it only involves (next-)nearest neighbor interactions in 4-dim or 5-dim lattice.
- Iterative algorithms (conjugate gradient, Lanczos, etc.) are used, which involve the matrix-vector multiplication.
- CUDA kernels can be optimized for the matrix-vector multiplication in QCD.

Lattice QCD

Kenneth G. Wilson
Nobel Prize (1982)

The QCD action $S = S_G(U) + \bar{\psi} D(U) \psi$

where $S_G(U)$ is the action of the gluon fields

$$\bar{\psi} D(U) \psi \equiv \bar{\psi}^f_{a\alpha x} D_f(U)_{a\alpha x, b\beta y} \psi^f_{b\beta y}$$

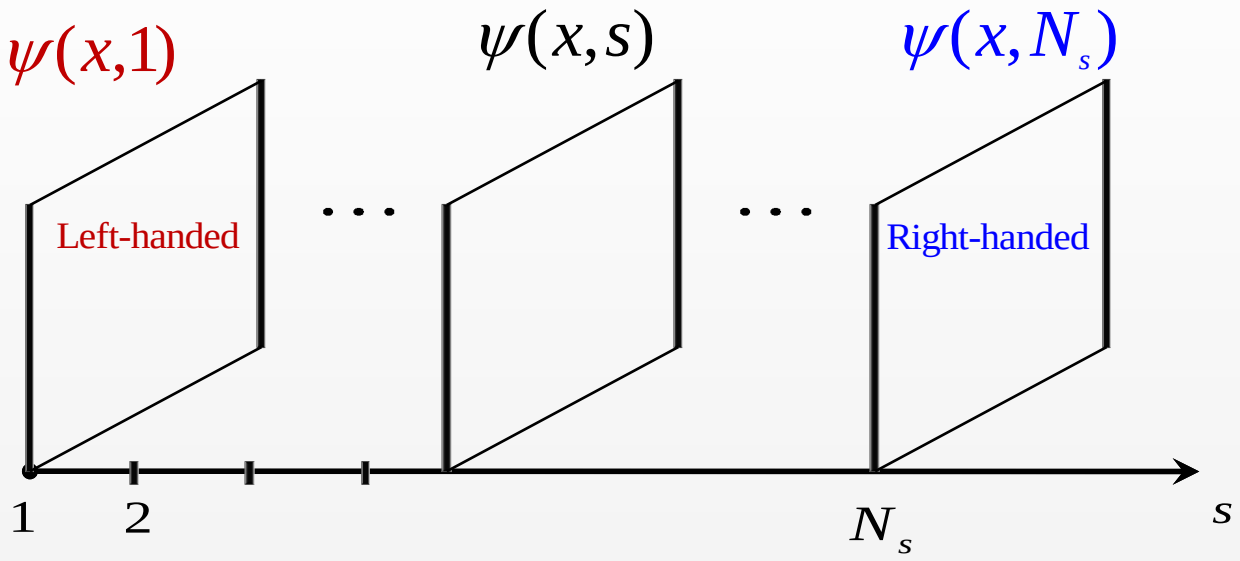
$f = u, d, s, c, b, t$	flavor index
$a, b = 1, 2, 3$	color index
$\alpha, \beta = 1, 2, 3, 4$	Dirac index
$x, y = 1, \cdots, N_{\text{sites}} = N_x N_y N_z N_t$	site index

For example, on the $16^3 \times 32$ lattice, for each flavor,

D is a complex matrix of size $1{,}572{,}864 \times 1{,}572{,}864$

$$\langle \mathcal{O}(\bar{\psi}, \psi, U) \rangle = \frac{\int dU d\bar{\psi} d\psi \mathcal{O}(\bar{\psi}, \psi, U) e^{-S}}{\int dU d\bar{\psi} d\psi e^{-S}} = \frac{\int dU \Theta(D^{-1}, U) \det(D) e^{-S_G}}{\int dU \det(D) e^{-S_G}}$$

Domain-Wall Fermions



D_{dwf} is a local op. with the nearest neighbor coupling along $\hat{s}$

$$\int [d\bar{\psi}][d\psi] \exp\left(-\bar{\Psi} D_{\text{dwf}} \Psi\right) = \det D_c \qquad D_c = \frac{1+\gamma_5 S}{1-\gamma_5 S}$$

$$N_s \to \infty, \quad S \to \frac{H}{\sqrt{H^2}}, \quad D_c\gamma_5 + \gamma_5 D_c = 0, \text{ Exact Chiral Sym.}$$

DWF with even-odd preconditioning

$$\mathcal{D}(m_q) = S_1^{-1} \begin{pmatrix} 1 & M_5 D_w^{\mathrm{EO}} \\ M_5 D_w^{\mathrm{OE}} & 1 \end{pmatrix} S_2^{-1}$$

Schur decomposition

$$\mathcal{D}(m_q) = S_1^{-1} \begin{pmatrix} 1 & 0 \\ M_5 D_w^{\mathrm{OE}} & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 0 & C \end{pmatrix} \begin{pmatrix} 1 & M_5 D_w^{\mathrm{EO}} \\ 0 & 1 \end{pmatrix} S_2^{-1}$$

$$C \equiv 1 - M_5 D_w^{\mathrm{OE}} M_5 D_w^{\mathrm{EO}}$$

For 2-flavors QCD, the pseudofermion action is

$$A_{PF} = \phi^\dagger C_{PV}^\dagger (C C^\dagger)^{-1} C_{PV} \phi \qquad C_{PV} \equiv C(m_q = 2m_0)$$

Conjugate Gradient Method

- Conjugate Gradient is an iterative method for solving the inverse of a sparse positive-definite Hermitian matrix.

$$Ax = b, \quad A = CC^\dagger$$

$x_0 :=$ initial guess

$r_0 := b - Ax$

$p_0 := r_0$

Iteration to convergence

$$\alpha_k = \frac{(r_k, r_k)}{(p_k, Ap_k)} = \frac{(r_k, r_k)}{(C^\dagger p_k, C^\dagger p_k)}$$

$$r_{k+1} = r_k - \alpha_k A p_k$$

$$\beta_{k+1} = \frac{(r_{k+1}, r_{k+1})}{(r_k, r_k)}$$

$$x_{k+1} = x_k + \alpha_k p_k$$

$$p_{k+1} = r_{k+1} + \beta_{k+1} p_k$$

CG is used for calculating fermion force, which is the most time-consuming part in the HMC simulation.

Mixed-Precision CG (1)

Single-precision operations are much faster then double-precision ones on GPU

High precision

$$\hat{A}\hat{x} = \hat{b}, \quad \hat{A} = \hat{C}\hat{C}^{\dagger}$$

$\hat{x} :=$ initial guess

$\hat{r} := \hat{b} - \hat{A}\hat{x}$

`while` $(|\hat{r}|^2 > \hat{\epsilon})\{$

$r := \hat{r}$

$p := \hat{r}$

$x := 0$

Low-precision CG

$\hat{x} := \hat{x} + x$

$\hat{r} := \hat{b} - \hat{A}\hat{x}$

$\}$

Low precision

$$Ax = r(= \hat{r}), \quad A = CC^{\dagger}$$

$\rho := (r, r)$

`while` $(\beta_0 > \epsilon)\{$

$v_0 := C^{\dagger}p$

$\alpha := \rho/(v_0, v_0)$

$r := r - \alpha C v_0$

$\rho' := \rho$

$\rho := (r, r)$

$x := x + \alpha p$

$p := r + (\rho/\rho')p$

$\}$

Mixed-Precision CG (2)

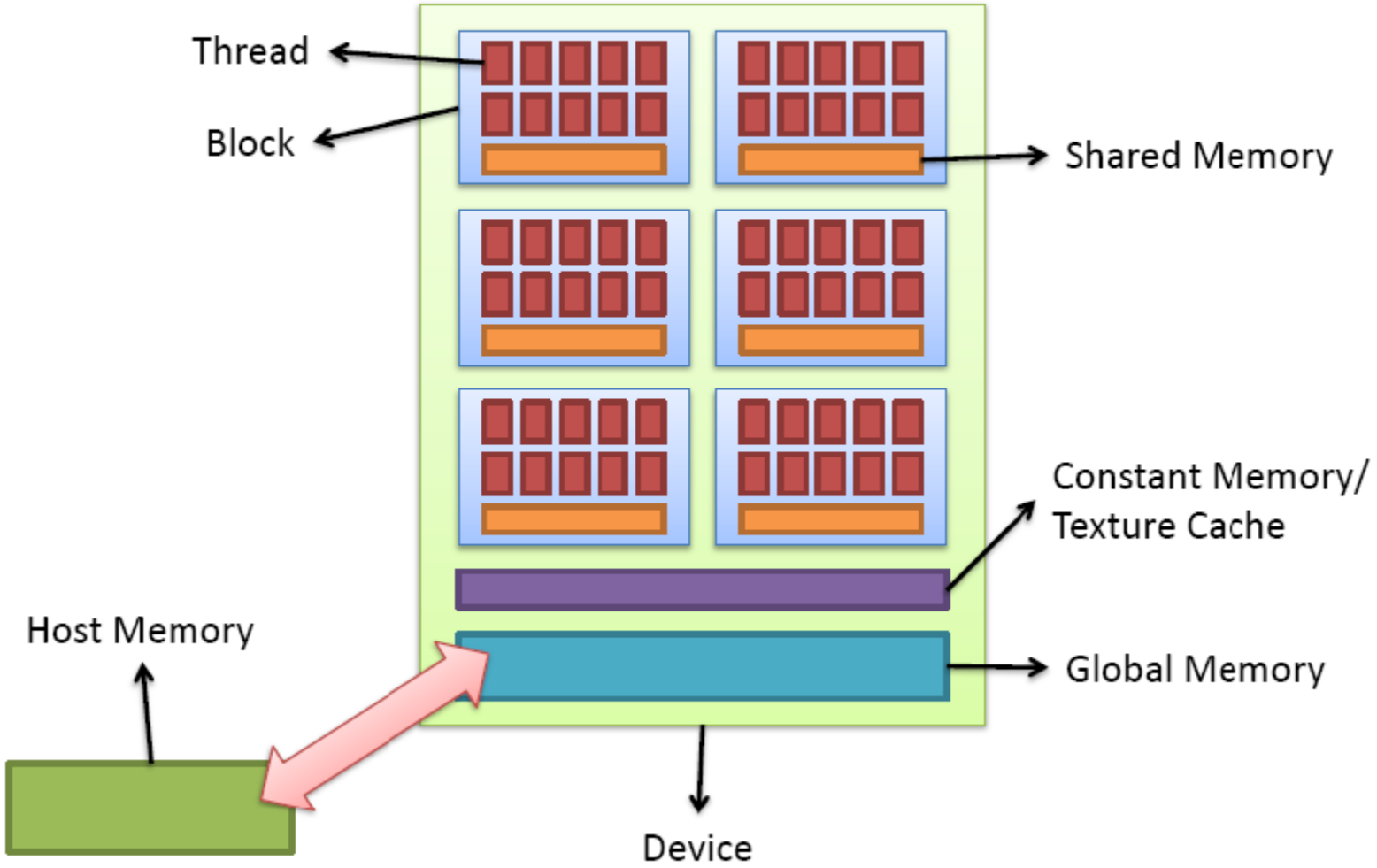
1. $r_k = b - Ax_k$
2. If $|r_k| < \varepsilon |b|$, then stop
3. Solve $At_k = r_k$ in single precision to an accuracy $\varepsilon_1 < 1$
4. $x_{k+1} = x_k + t_k$
5. Go to 1.

Proof :

Let $u_k = r_k - At_k$, $\quad |u_k| < \varepsilon_1 |r_k|$

then $|r_{k+1}| = |b - Ax_{k+1}| = |b - Ax_k - At_k| = |u_k| < \varepsilon_1 |r_k| < |r_k|$

CUDA Architecture



CG Kernels Overview (single-prec.)

$$C \equiv 1 - M_5 D_w^{\mathrm{OE}} M_5 D_w^{\mathrm{EO}}$$

The multiplication of M5 and Dw are implemented in different kernels.

Each line below is implemented in one kernel.

$$v_0 := C^\dagger p$$

$$\alpha := \rho/(v_0, v_0)$$

$$r := r - \alpha C v_0$$

$$\rho' := \rho$$

$$\rho := (r, r)$$

$$x := x + \alpha p$$

$$p := r + (\rho/\rho')p$$

$$v_1 := M_5^\dagger p$$

$$v_0 := (D_w^{\mathrm{OE}})^\dagger v_1$$

$$v_1 := M_5^\dagger v_0$$

$$v_0 := p - (D_w^{\mathrm{EO}})^\dagger v_1$$

$$\alpha := \rho/(v_0, v_0)$$

$$v_1 := D_w^{\mathrm{EO}} v_0, \quad r := r - \alpha v_0$$

$$v_0 := M_5 v_1$$

$$v_1 := D_w^{\mathrm{OE}} v_0$$

$$r := r + \alpha M_5 v_1$$

$$\rho' := \rho, \quad \rho := (r, r)$$

$$x := x + \alpha p, \quad p := r + (\rho/\rho')p$$

CG Kernels Overview (double-prec.)

$$A \equiv CC^\dagger$$
$$C \equiv 1 - M_5 D_w^{\mathrm{OE}} M_5 D_w^{\mathrm{EO}}$$

The multiplication of M5 and Dw are implemented in different kernels.

$$\hat{x} := \hat{x} + x$$
$$\hat{r} := \hat{b} - \hat{A}\hat{x}$$

Each line below is implemented in one kernel.

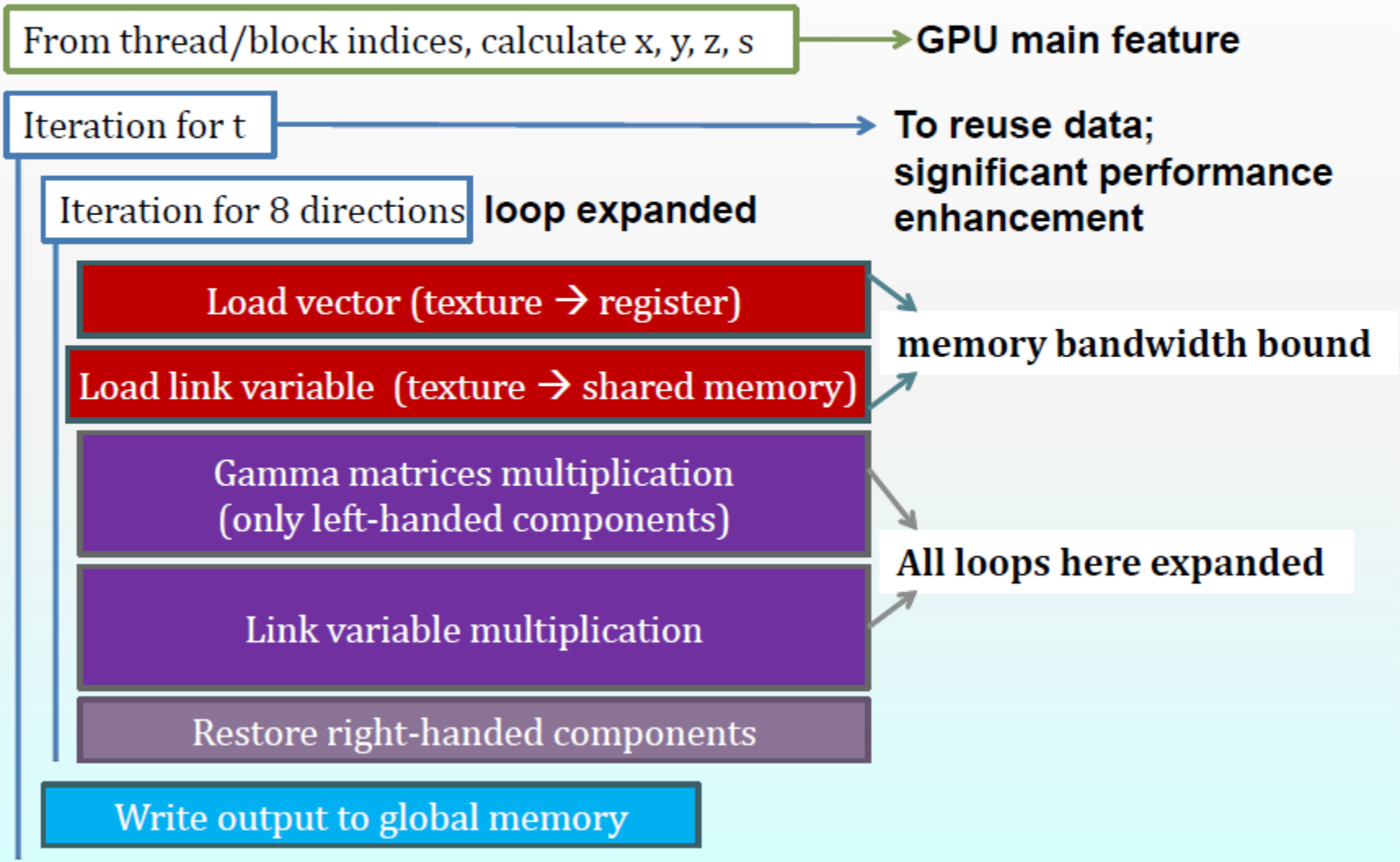
$$v_1 := M_5^\dagger p$$
$$v_0 := (D_w^{\mathrm{OE}})^\dagger v_1$$
$$v_1 := M_5^\dagger v_0$$
$$v_0 := (D_w^{\mathrm{EO}})^\dagger v_1$$
$$v_1 := p - v_0$$
$$v_2 := D_w^{\mathrm{EO}} v_1$$
$$v_0 := M_5 v_2$$
$$v_2 := D_w^{\mathrm{OE}} v_0$$
$$v_1 := v_1 - v_2$$
$$r := b - v_1$$

CG Kernels (Dw multiplication)

$$(D_w^{\mathrm{OE}})_{xx'} = -\frac{1}{2}\sum_{\mu}\left[(1-\gamma_\mu)U_\mu(x)\delta_{x+a\hat{\mu},x'} + (1+\gamma_\mu)U_\mu^\dagger(x')\delta_{x-a\hat{\mu},x'}\right]$$

- **Hopping terms**
  - Texture is used for caching data
  - Internal loop is used to reuse read-in data
- **Link variables multiplication**
  - For a given μ, U is the same for all s
    → use shared memory
- **Gamma matrices multiplication**
  - Only left-handed Dirac indices are calculated

Dw multiplication Implementation



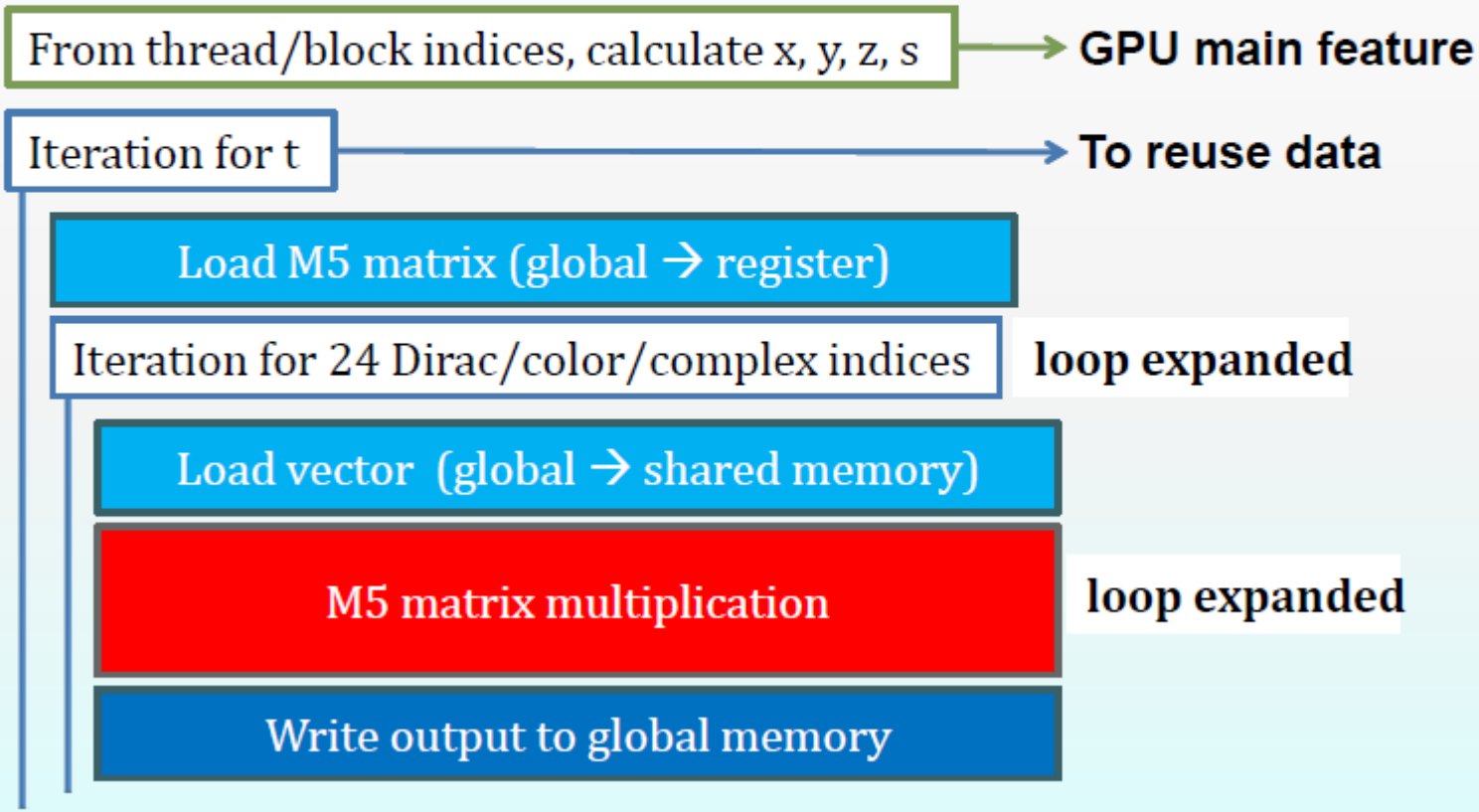
CG Kernels (M5 multiplication)

$$M_5 = \left[(d - m_0) + \omega^{-1/2}(1 - L)(1 + cL)^{-1}\omega^{-1/2}\right]^{-1}$$

- Block diagonal in the chiral basis.
- Does not depend on *x*, *y*, *z*, *t*, or color index.
- It is **a constant matrix-vector** multiplication in the 5th-dim space.
- Use shared memory for storing source vector

$$v_{s'} = \sum_s (M_5)_{s's} v_s$$

M5 multiplication Implementation



CG Kernels (More Tunings)

- Try to reuse data as much as possible!
- When doing parallel reduction (calculating norm), do partly in the previous kernel:

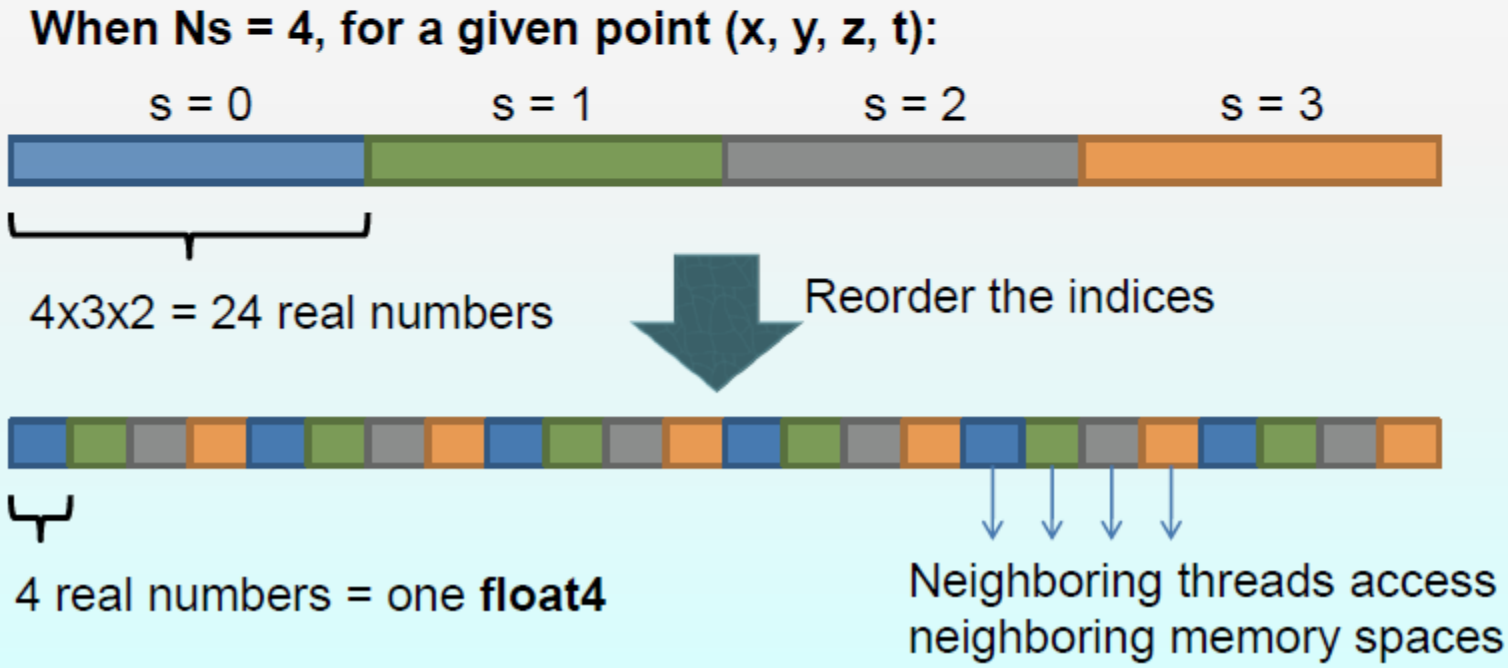
$v_0 := p - (D_w^{\mathrm{EO}})^\dagger v_1$ → Do a "pre-parallel reduction" within each block

$\alpha := \rho/(v_0, v_0)$ → Parallel reduction of v_0

- Addition/subtraction: try to combine these simple operations with existing multiplication kernels, for examples: $v_1 := D_w^{\mathrm{EO}} v_0, \quad r := r - \alpha v_0$

Memory Management (2)

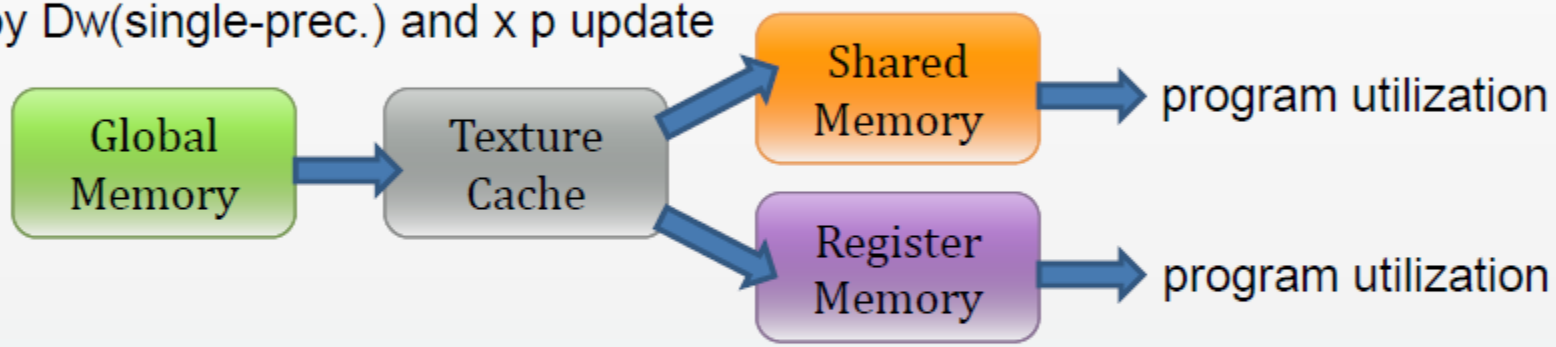
- Reorder array indices such that adjacent threads will access adjacent memory spaces.
  → better coalesce!



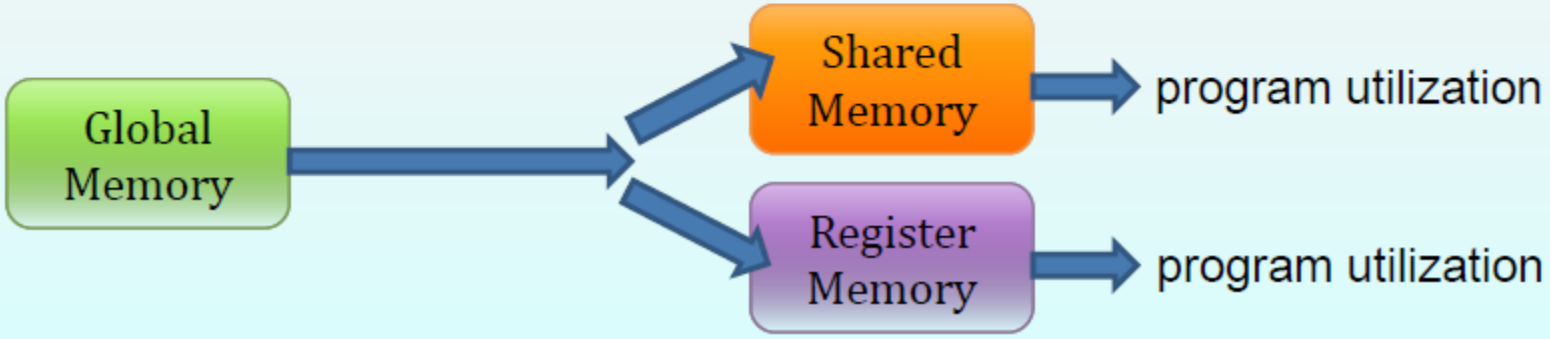
Memory Management (3)

- Use texture and shared memory

Used by Dw(single-prec.) and x p update



Used by M5(single-prec.), and all double prec. kernels



Benchmarks

- CG (mixed prec.) attains 317 Gflops on GTX580

	Dw(Single)	M5(Single)	Dw(Double)	M5(Double)	CG(Mixed)
GTX285	177	346	33	69	181
C1060	128	290	29	61	132
C2070	171	244	22	96	156
GTX480	293	309	37	116	252
GTX580	338	445	41	150	317

All numbers are in unit of Gflops, tested with ODWF on $16^3 \times 32 \times 16$ lattice

- The bottleneck is Dw single-precision multiplication

GPU Cluster at NTU

- 280 Nvidia GPUs with peak performance > 300 TFLOPS
- 22/36/7 Nvidia C2070/GTX580/GTX480 (total 47 GPUs), connected to 27 servers (total 27 Intel i7)
- 16 units of Nvidia Tesla S1070 (total 64 GPUs) connected to 16 servers (total 48 Intel QC Xeon)
- 32 Nvidia C1060 (total 32 GPUs), connected to 16 servers (total 16 Intel i7)
- 122 Nvidia GTX285 (total 122 GPUs), connected to 62 servers (total 62 Intel i7)
- Hard disk storage > 300 TB, Lustre cluster file system
- Developed efficient CUDA codes for full QCD. 320/**156/180/132 Gflops** for **GTX580/C2070/GTX285/C1060**
- Attaining **80 TFLOPS (sustained) for LQCD with Optimal DWF**

GPU Cluster at NTU (a partial view)

Conclusions and Outlook

- GPU has provided the optimal price/performance for large-scale lattice QCD simulations.
- We have implemented efficient CUDA codes for lattice QCD with domain-wall fermion. On GTX580, our CG solver attains 320 Gflops (sustained).
- Lattice QCD with domain-wall fermion on the $24^3 \times 48 \times 16$ can be simulated efficiently with one single GPU, and for larger lattices (e.g., $32^3 \times 64 \times 16$) with multiGPUs.
- GPU has revolutionized the advancement of QCD

Conclusions and Outlook (cont.)

- First Physics Results from the NTU GPU cluster:

1. T.W. Chiu, T.H. Hsieh, Y.Y. Mao (TWQCD),
   "Topological Susceptibility in Two Flavors Lattice QCD with the Optimal Domain-Wall Fermion",
   Phys. Lett. B 702 (2011) 131.

2. T.W. Chiu, T.H. Hsieh, Y.Y. Mao (TWQCD),
   "Pseudoscalar Meson in Two Flavors QCD with the Optimal Domain-Wall Fermion",
   arXiv:1109.3675

Backup slides

Quantum Chromodynamics (QCD)

The quantum field theory for the strong interaction between quarks and gluons which build up the hadrons (e.g., neutron, proton, pion, etc.). QCD provides the framework to understand the nuclear force/energy from the first principles, and plays an important role in the evolution of the early universe, from the quark gluon "plasma" phase to the hadron phase.

Salient features:

- Gauge group $SU(3) \Rightarrow$ gluons have self-interactions.
- Asymptotic freedom: $g(r) \to 0$ as $r \to 0$.
- IR slavory: $g(r) \approx 1$ at $r \approx 1$ fm $\Rightarrow$ quark (color) confinement
- Spontaneously chiral symmetry breaking

- It took 23 years (1974 ~1997) to realize that **Lattice QCD with Exact Chiral Symmetry** is the ideal theoretical framework to study the nonperturbative physics from the first principles of QCD.

- **But, it is challenging to perform the simulation** such that the chiral sym. is preserved to very high precision & all topological sectors are sampled ergodically.

- Since 2009, the **TWQCD** collaboration has been simulating QCD with **optimal domain-wall quarks**. The chiral sym. is preserved to a good precision with $m_{res}a \approx 0.0004$, and all topological sectors are sampled ergodically.

Exact Chiral Symmetry on the Lattice

The proper way to break the chiral symmetry at finite lattice spacing is to impose the Ginsparg-Wilson relation (1982)

$$D\gamma_5 + \gamma_5 D = D\gamma_5 D$$

equivalently, $D^{-1}(x,y)\gamma_5 + \gamma_5 D^{-1}(x,y) = \gamma_5 \delta_{x,y}$

which is realized by the Domain-Wall Fermion (Kaplan,1992), and the overlap Dirac operator (Neuberger,1998)

$$D = \left(I + \gamma_5 \frac{H}{\sqrt{H^2}} \right), \quad H^\dagger = H,$$

D is exponentially local for sufficently smooth gauge field.

In the continuum limit $a \to 0$, $D \to \gamma^\mu(\partial_\mu + igA_\mu)$.

Central Problems in Lattice QCD

- To compute the quark propagator D^{-1}
- To compute the (low-lying) eigenmodes of D

The matrix D is prohibitively large for exact solvers.
Iterative algorithms involve the matrix-vector multiplication.
For the overlap fermion operator, it involves

$$\frac{H}{\sqrt{H^2}} \cdot Y$$

The inverse square-root cannot be computed exactly.
What is the best way to proceed ?

Nested Conjugate Gradient

To compute quark propagator requires nested CG

$$D \cdot Y \equiv \left(I + \gamma_5 \frac{H}{\sqrt{H^2}} \right) \cdot Y = |b\rangle$$

$$\frac{H}{\sqrt{H^2}} \cdot Y = H R^{(n-1,n)}(H^2) \cdot Y$$

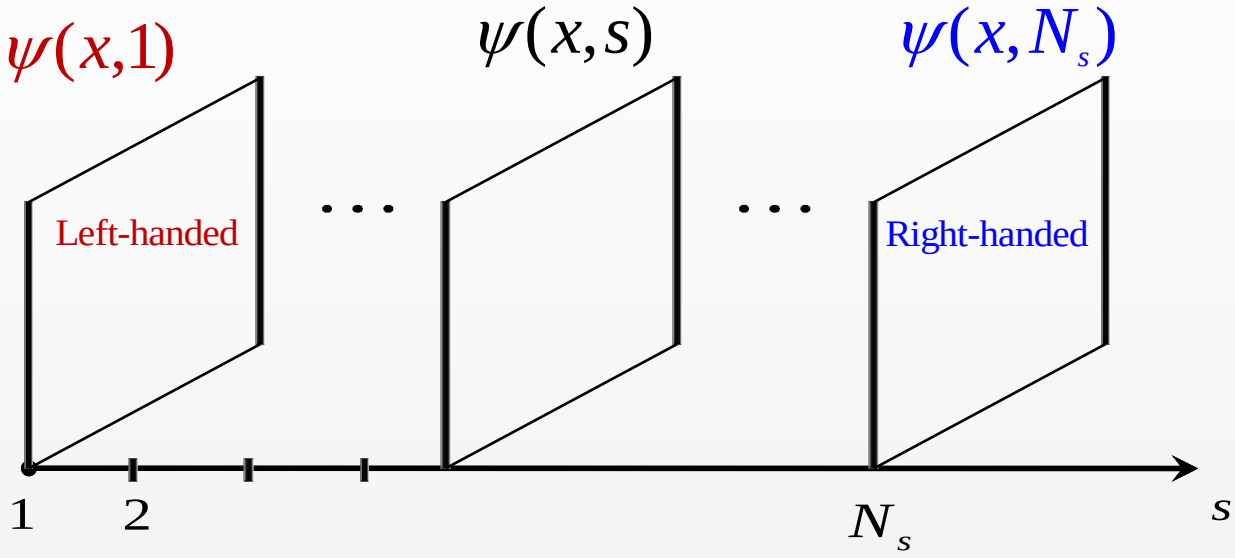
$$R^{(n-1,n)}(H^2) \cdot Y = \sum_{l=1}^{n} \frac{b_l}{H^2 + d_l} Y = \sum_{l=1}^{n} b_l Z^{(l)}$$

$\left(H^2 + d_l \right) Z^{(l)} = Y$ solved by CG with multi-shifts

Q: Can we avoid the inverse square root ?

A: Yes, to introduce an extra dim. (degree of freedom).

Domain-Wall Fermions



D_{dwf} is a local op. with the nearest neighbor coupling along $\hat{s}$

$$\int [d\bar{\psi}][d\psi] \exp\left(-\bar{\Psi} D_{\text{dwf}} \Psi\right) = \det D_c \qquad D_c = \frac{1+\gamma_5 S}{1-\gamma_5 S}$$

$$N_s \to \infty, \quad S \to \frac{H}{\sqrt{H^2}}, \qquad D_c\gamma_5 + \gamma_5 D_c = 0, \text{ Exact Chiral Sym.}$$

For finite N_s, which DWF gives the best rational approx. to S ?

Optimal Domain-Wall Fermion

[TWC, Phys. Rev. Lett. 90 (2003) 071601]

$$A_{\text{odwf}} = \sum_{s,s'=1}^{N_s} \sum_{x,x'} \bar{\psi}_{x,s} \left[\left(I + \omega_s D_w \right)_{x,x'} \delta_{s,s'} - \left(I - \omega_s D_w \right)_{x,x'} \left(P_- \delta_{s',s+1} + P_+ \delta_{s',s-1} \right) \right] \psi_{x',s'}$$

$$\equiv \bar{\Psi} D_{\text{odwf}} \Psi$$

$$D_w = \sum_{\mu=1}^{4} \gamma_\mu t_\mu + W - m_0, \qquad m_0 \in (0,2)$$

$$t_\mu(x,x') = \frac{1}{2}\left[U_\mu(x) \delta_{x',x+\mu} - U_\mu^\dagger(x') \delta_{x',x-\mu} \right]$$

$$W(x,x') = \sum_{\mu=1}^{4} \frac{1}{2}\left[2\delta_{x,x'} - U_\mu(x) \delta_{x',x+\mu} - U_\mu^\dagger(x') \delta_{x',x-\mu} \right]$$

with boundary conditions

$$P_+ \psi(x,0) = -\frac{m_q}{2m_0} P_+ \psi(x,N_s), \qquad m_q: \text{ bare quark mass}$$

$$P_- \psi(x,N_s+1) = -\frac{m_q}{2m_0} P_- \psi(x,1), \qquad P_\pm = \frac{1}{2}(1 \pm \gamma_5)$$

The state-of-the-art in the simulations of unquenched QCD with exact chiral symmetry

- RBC and UKQCD Collaborations

  Machine: QCDOC

  Lattice fermion: Domain-Wall Fermion

  Lattice sizes: $16^3 \times 32 \times 16,\ 24^3 \times 48 \times 16,\ 32^3 \times 64 \times 16$

- JLQCD Collaboration

  Machine: IBM BlueGene/L

  Lattice fermion: Overlap Fermion (with fixed topology $Q_t = 0$)

  Lattice sizes: $16^3 \times 32,\ 16^3 \times 48$

- TWQCD Collaboration

  Machine: **GPU cluster with 280 GPUs**

  Lattice fermion: Optimal Domain-Wall Fermion

  Lattice sizes: $16^3 \times 32 \times 16,\ 20^3 \times 40 \times 16,\ 24^3 \times 48 \times 16$

Memory Management (1)

- Basics of the memory hierarchy:

	Size	Access	Bandwidth
Global	Large	r/w by all threads and host	Slow
Constant	Small	Read only by all threads	Fast
Texture	Small cache	Read only by all threads	Fast
Shared	Very small	r/w by all threads within one block	Very fast
Register	Very small	r/w by only one thread	Very fast

  - Shared memory may have bank conflicts
  - Texture can take care of the locality.
- GPU computing is memory bandwidth bound!

Thread/Block Management (1)

- Basic ideas of the execution mode:
  - Parallelize a loop by designating the value of the *loop counter* to each thread.
  - Try to have as many threads as possible, which means more threads working at the same time.
  - All threads inside the same block can access (r/w) the shared memory.

Thread/Block Management (2)

- Number of threads per block
  - Should be tested to find the best value (may be limited by resource in one block)
  - Must be a multiple of **half-warp**.
- memory bandwidth bound → try to reuse data.
  - Larger number of blocks does NOT necessarily mean better performance!
  - Using *loop* inside kernel to reduce the number of blocks sometimes runs faster.
  - For Dw multiplication, *loop* can further help to reuse one of the hopping data.