Recent algorithm and machine developments for lattice QCD

Ken-Ichi Ishikawa (Hiroshima Univ.)

Lattice 2008,July 16

1. Plan of My Talk

2. Machine trends
   - New machines
     - BG/P, T2K, QPACE project, Pet-Ape project.
   - Many cores
     - GPGPU CUDA
3. Algorithmic developments for dynamical QCD (Wilson type)
   - HMC with
     - Preconditioning for HMC action and UV/IR separation: Domain-Decomposition,RHMC, Schur complement…
     - Multiple timescale MD integrator
   - Solver with
     - Inner-Outer(mixed prec.), Deflation, Adoptive Multi Grid.
4. Outlook: Physics at 1Pflops
   - Finer lattice (continuum limit or charm quark)
   - Larger volume (multi hadron system)

I apologize to everyone if whose work is not properly cited.

2. Machine Trends

- New machines
  - Blue Gene/P
    - Successor of QCDSP, QCDOC, Blue Gene/L

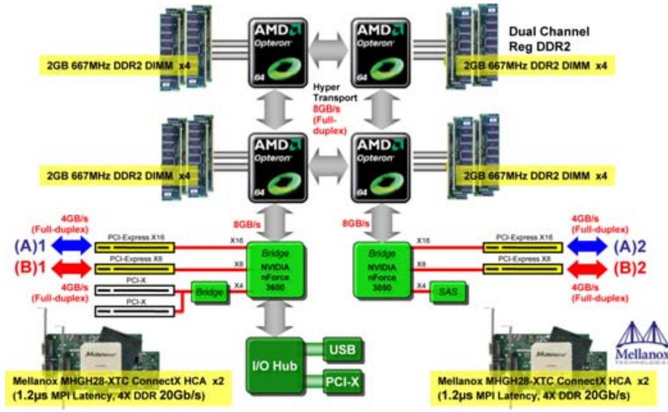
      [P.Boyle et al., IBM J. Res. and Dev. 49 (2005)
      http://www.research.ibm.com/journal/rd/492/boyle.html]
    - 4Way SMP PowePC@0.85GHz
    - Scalable 3D torus network
    - Population is incleasing
    - Thin node / O(100,000) Many nodes
    - Byte/Flop balanced
    - Fine grained parallelization.

2. Machine Trends (cont'd)

- New machines
  - T2K open super computer project (Tsukuba-Tokyo-Kyoto)

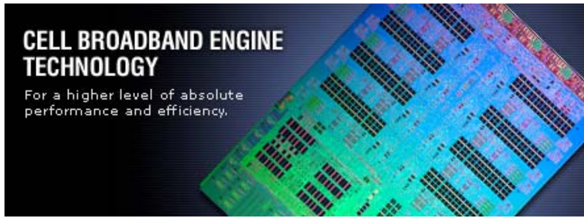
[http://www.open-supercomputer.org/]

    - 4 Way Opteron (Barcelona) node cluster (commodity base).
    - 648nodes@tsukuba, 147GFlops/node (Fat node)
    - Quad core, 4 way
    - Multi-rail fat tree network
    - Many core / Fat node / O(1,000) few nodes
    - Maintain Byte/Flop at each level
    - Data Blocking is required



2. Machine Trends (cont'd)

- New machines (for QCD)
  - QPACE project (QCD PArallel computing on the CEll/B.E.) 2008-2009 [Poster by A. Nobile "Status of the QPACE Project"]
    - Fund by Deuche Forschungsgemeinscaft (DFG)
    - Collaboration with IBM Germany.
    - Dedicated for LQCD. 200TFlops (2009)
    - Cell Broadband Engine cluster. [PowerXCell 8i, 102GFlops(DP)]
    - Custom 3D torus Scalable network (FPGA)
    - Low power consumption 1.5W/GFlops
    - Many core / Fat node / O(1,000) Few nodes
    - Maintain Byte/Flop at each level
    - Data Blocking is required


CELL BROADBAND ENGINE TECHNOLOGY
For a higher level of absolute performance and efficiency.

QCD with CELL: Spary,Hill,Trew hep-lat/0804.3654;
S.Motoki & A. Nakamura Lat2007;
F.Belletti et al. LAT2007

2. Machine Trends (cont'd)

- New machines
  - **Pet-APE** project (Petaflops Array Processor Experiment)

    [INFN APE Groupe, Italy, to apper in NUOVO CHIMENTO]

    EMAIL From Davide Rossetti@ROMA1.INFN

    - Successor of APEmille, apeNEXT.
    - Reference computing platform for LQCD (2009-2014)
    - Custom CPU: Apotto
    - Custom Network ApeNet+ 3D Torus.
    - Aiming for good price/performance.
    - Thin node / O(100,000) Many nodes ?

2. Machine Trends (cont'd)

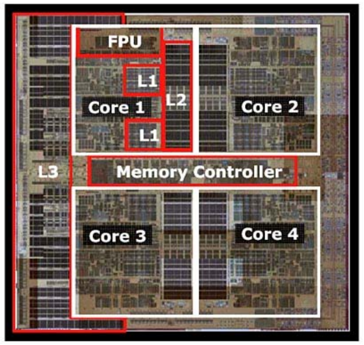
- Many cores (QPACE, T2K)

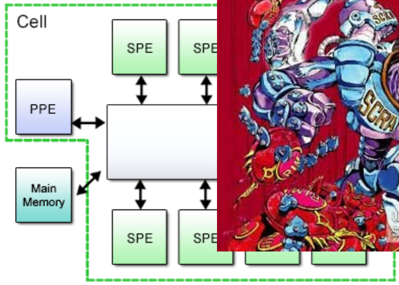
To make use of the full machinery of many transistors on a chip, many core architecture is employed for recent processor

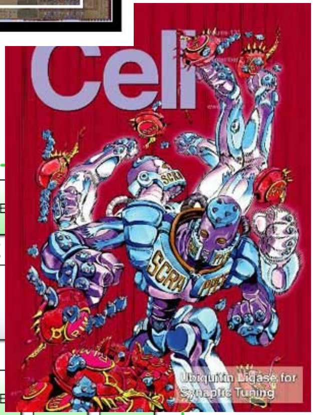
  - Intel: Core 2 Quad (4cores, 3GHz, 48GFlops),…
  - AMD: Phenom (4cores, 2.4GHz, 38GFlops), …
  - IBM: Power X Cell 8i (1+8cores, 3.2GHz, 102GFlops)
  - SUN: UltraSparc T2 (8cores)

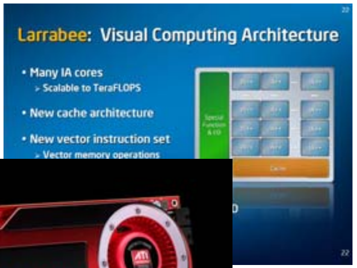
The trend is 8 cores, 16 cores,…., many cores

  - Intel larrabee 80 cores?
  - AMD/ATI GPGPU firestrem 800 cores?
  - NVIDIA GPGPU CUDA 240 cores?
  - As a many core example , GPGPU





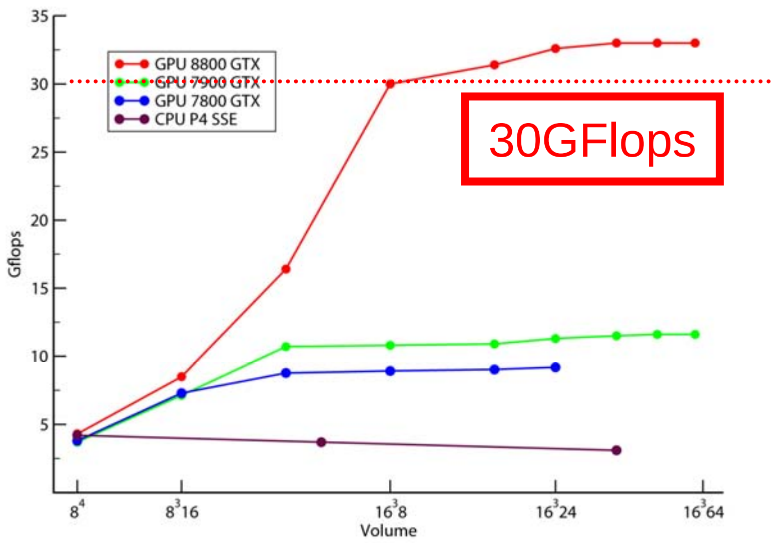



Larrabee: Visual Computing Architecture

- Many IA cores
  - Scalable to TeraFLOPS
- New cache architecture
- New vector instruction set
  - Vector memory operations

2. Machine Trends (cont'd)

- GPGPU
  - **"Lattice QCD as a video game"**, G.I.Egri, Z.Fodor, S.D.Katz, D.Nogradi, K.K.Szabo, hep-lat/0611022.
    - NVIDIA G80 arch. > 300 GFlops(SP)
    - Lattice Wilson kernel > 30 GFlops
    - Difficult to program using Graphic API (OpenGL)



  - NVIDIA provides HPC GPGPU language
    - CUDA (a C/C++ simple extension)
    - Easy to learn, but requires hardware/memory model knowledge

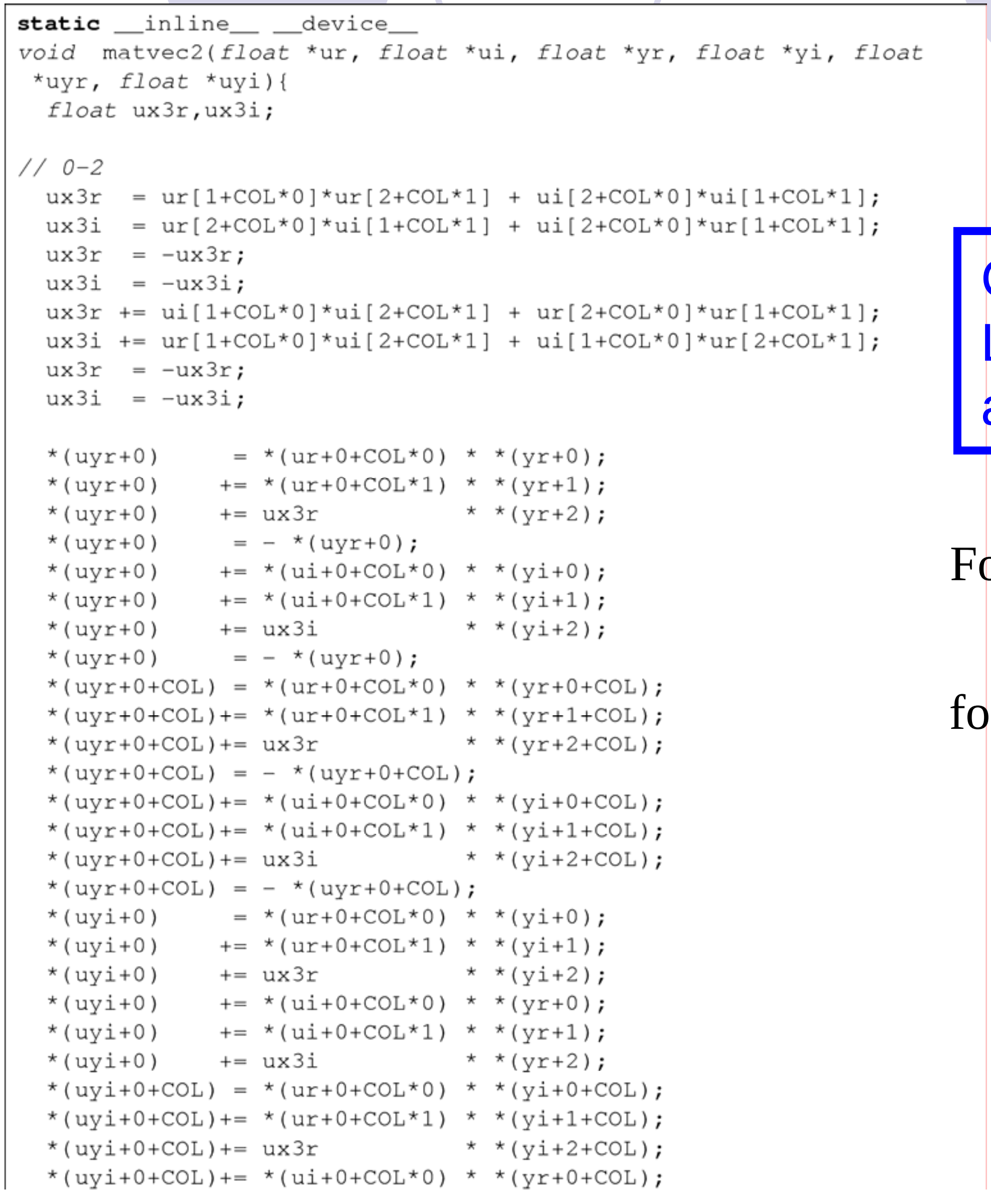
[Poster by C. Rebbi, "Blastign Through Lattice Calc. using CUDA"
talk by F. Di Renzo, "GPU computing for 2-d spin systems:CUDA vs OpenGL"]

  - My experience with CUDA (GeForce 8800 GTX)
    [**NO WARRANTY CUDA code:**http://theo.phys.sci.hiroshima-u.ac.jp/~ishikawa/CUDA/CudaQCDSolver_0.06.tar.gz]
    - Hopping matrix mult (16^4) can also achieve > 40 GFlops.

C. Rebbi (Poster):
Wilson Dirac 62GFlops!
with Nvidia Tesla C870

2. Machine Trends (cont'd)

- My experience with CUDA (GeForce 8800 GTX)



```
static __inline__ __device__
void  matvec2(float *ur, float *ui, float *yr, float *yi, float
 *uyr, float *uyi){
  float ux3r,ux3i;

// 0-2
  ux3r  = ur[1+COL*0]*ur[2+COL*1] + ui[2+COL*0]*ui[1+COL*1];
  ux3i  = ur[2+COL*0]*ui[1+COL*1] + ui[2+COL*0]*ur[1+COL*1];
  ux3r  = -ux3r;
  ux3i  = -ux3i;
  ux3r += ui[1+COL*0]*ui[2+COL*1] + ur[2+COL*0]*ur[1+COL*1];
  ux3i += ur[1+COL*0]*ui[2+COL*1] + ui[1+COL*0]*ur[2+COL*1];
  ux3r  = -ux3r;
  ux3i  = -ux3i;

  *(uyr+0)      = *(ur+0+COL*0)  * *(yr+0);
  *(uyr+0)     += *(ur+0+COL*1)  * *(yr+1);
  *(uyr+0)     += ux3r           * *(yr+2);
  *(uyr+0)      = - *(uyr+0);
  *(uyr+0)     += *(ui+0+COL*0)  * *(yi+0);
  *(uyr+0)     += *(ui+0+COL*1)  * *(yi+1);
  *(uyr+0)     += ux3i           * *(yi+2);
  *(uyr+0)      = - *(uyr+0);
  *(uyr+0+COL)  = *(ur+0+COL*0)  * *(yr+0+COL);
  *(uyr+0+COL)+= *(ur+0+COL*1)  * *(yr+1+COL);
  *(uyr+0+COL)+= ux3r           * *(yr+2+COL);
  *(uyr+0+COL)  = - *(uyr+0+COL);
  *(uyr+0+COL)+= *(ui+0+COL*0)  * *(yi+0+COL);
  *(uyr+0+COL)+= *(ui+0+COL*1)  * *(yi+1+COL);
  *(uyr+0+COL)+= ux3i           * *(yi+2+COL);
  *(uyr+0+COL)  = - *(uyr+0+COL);
  *(uyi+0)      = *(ur+0+COL*0)  * *(yi+0);
  *(uyi+0)     += *(ur+0+COL*1)  * *(yi+1);
  *(uyi+0)     += ux3r           * *(yi+2);
  *(uyi+0)     += *(ui+0+COL*0)  * *(yr+0);
  *(uyi+0)     += *(ui+0+COL*1)  * *(yr+1);
  *(uyi+0)     += ux3i           * *(yr+2);
  *(uyi+0+COL)  = *(ur+0+COL*0)  * *(yi+0+COL);
  *(uyi+0+COL)+= *(ur+0+COL*1)  * *(yi+1+COL);
  *(uyi+0+COL)+= ux3r           * *(yi+2+COL);
  *(uyi+0+COL)+= *(ui+0+COL*0)  * *(yr+0+COL);
```

Cuda code example:
Link variable times 2-Spinor code
almost C language

For Single site data,

$$w, y : 2-\text{spinor},\ \ U : SU(3)\ \text{matrix}$$

for $\alpha = 1,2$ and $a = 1,2,3$

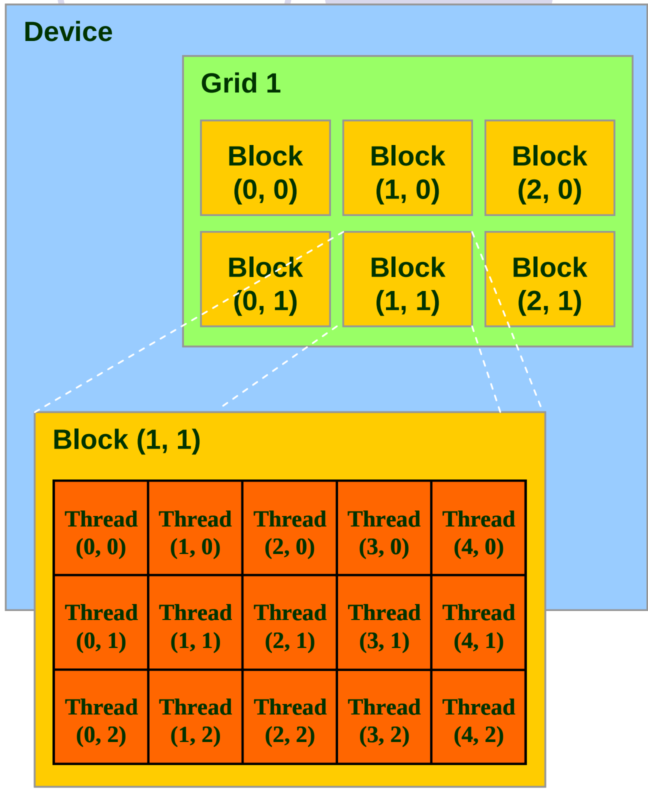
$$w(a,\alpha) = \sum_{b=1}^{3} U(a,b)\, y(b,\alpha)$$

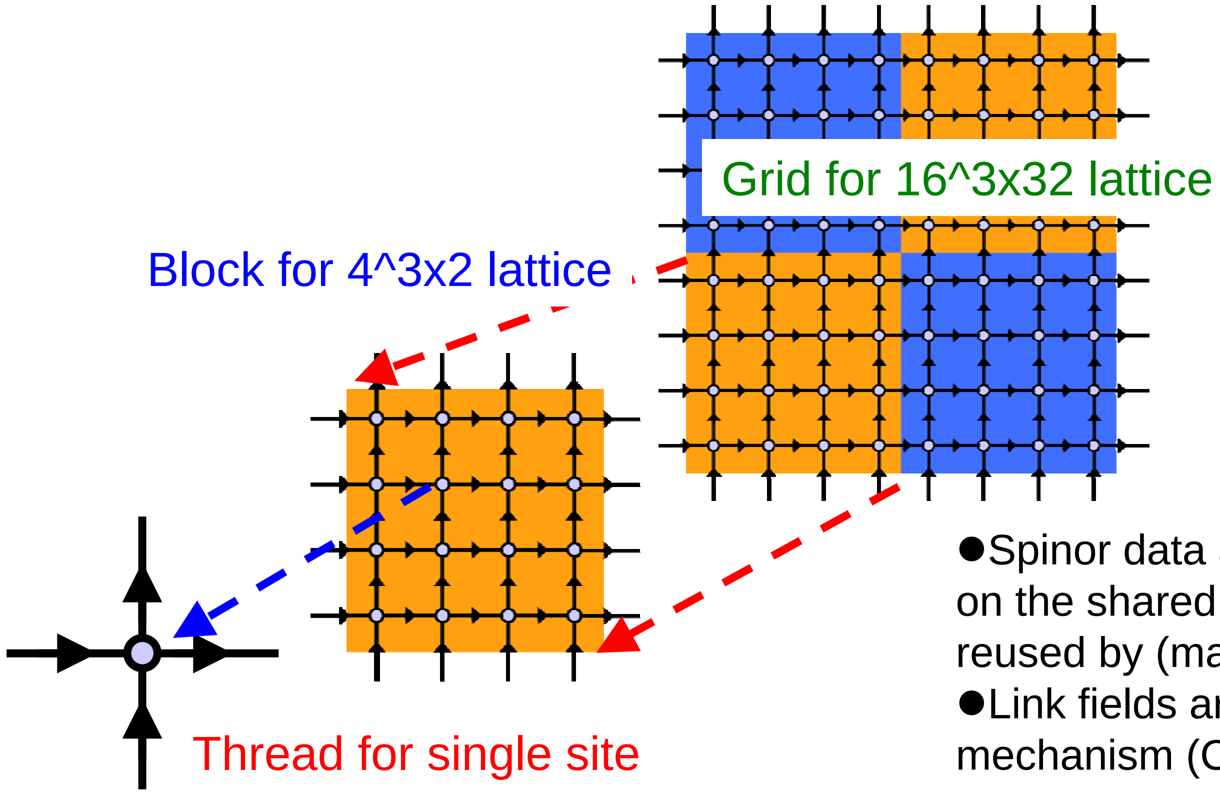
2. Machine Trends (cont'd)

- My experience with CUDA (GeForce 8800 GTX)

CUDA Programming model

- Single Program Multiple Data (SPMD)
- Nested threading. Grid / Block / Thread
- Thread ID + Block ID (Corresponds to MPI RANK)
- Block has local memory shared by threads in a block.





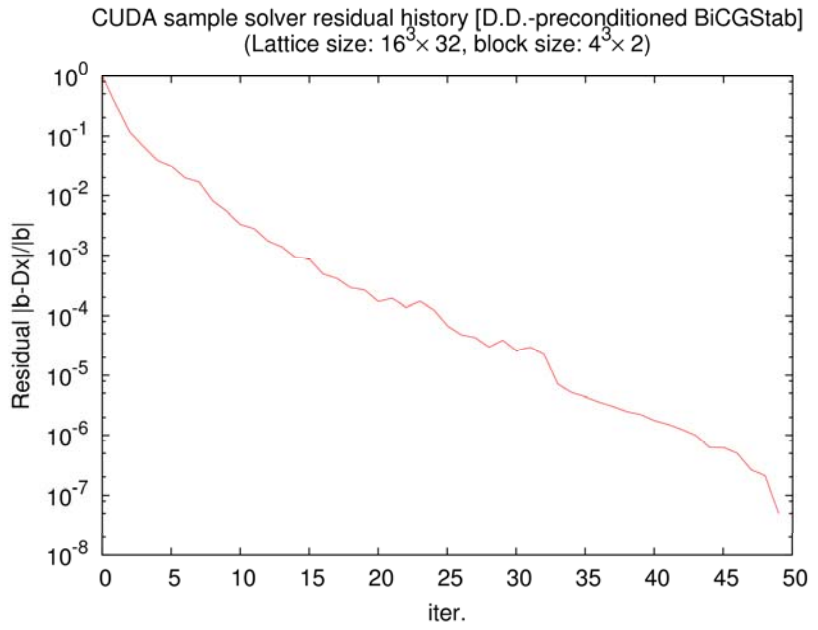
- Spinor data are vector loaded [100GByte/sec] on the shared memory on each block. They are reused by (max 8 times/ min 4 times).
- Link fields are loaded via Texture Fetching mechanism (Cached).

2. Machine Trends (cont'd)

- My experience with CUDA (GeForce 8800 GTX)

CUDA Language is Ready for Lattice QCD!!!

A test result with CUDA solver (Single precision)



●CELL has similar feature?

●How about other accelerator? (AMD/ATI card, ClearSpeed)

See also CUDA Works in This conference:
●F. Di Renzo, "GPU computing for 2-d spin sytems:CUDA vs OpenGL"
●C. Rebbi, "Blasting Through Lattice Calculations using CUDA"
CELL Works:
●V. Kindratenko, "Cell processor implementation of a MILC lattice QCD application"

2. Machine Trends (cont'd)

- GPGPU

  This year Nvidia and AMD/ATI provide DP enabled architecture
  - NVIDIA GT200 (Tesla 10series)
    - 240 SP (SP cores), 30 DP cores
    - ～1,000(or 600)Glops(SP), ～90GFlops(DP)

C. Rebbi (Poster): Wilson Dirac 100 GFlops! with Nvidia GTX280

  - AMD/ATI RV770 (Firestream 9250)
    - 640 SP units, (160 DP units?)
    - 1.2TFlops (SP), 200 GFlops (DP)
    - AMD Stream SDK

Product: AMD FireStream™ 9250
(Available September 2008)

Breaking the 1 TFLOPS barrier at under 150 watts!

  - For QCD
    - No ECC, check the result on the host side.
    - O(1000) thread programming/SIMD programming is required. (1site=1thread)
    - Make use of the Local memories attached each core for good efficiency.
    - Host ⇔device communication is limited by PCI-E x16 G2 speed (8GB/sec (sustained at 2GB/sec))

2. Machine Trends (cont'd)

- Thin node / O(100,000) nodes (BG/P , Pet-APE)
  - Uniform Fine Grained Parallelization is required.
  - 10GFlops/CPU, 100,000 nodes = 1PF

- Many cores / Fat node / O(1,000) nodes (T2K, QPACE, GPGPU)
  - Core/CPU/Node Hierarchy exists.
  - Data Bandwidth is not uniform.
  - Data blocking is required at each level.

- 1～10 PFlops machine trends?
  - My expectation is Many core/Fat node/O(1,000-10,000) nodes
  - Near future: Intel larrabee, CELL, GPGPU, …..
    - 200 GFlops/CPU , 8 CPU/node=1.6TF/node, 1,000 node=1.6PFlops
    - 1 TFlops/CPU, 4 CPU/node = 4TF/node, 1,000 node= 4PF

3. Algorithmic developments for dynamical QCD

- Recent improvement strategy (HMC)
  - Two key technologies for HMC algorithm

(1) Transform/split det[D] using preconditioner (Action Prec.)

UV/IR separation [de Forcrand, Takaishi, NPB(Proc.Suppl.)53,Lat96]

$D:$ Lattice Dirac op., $P:$ a preconditioner

choose P s.t. $\mathrm{cond}(DP) < \mathrm{cond}(D)$ and easy to compute $\det[P]$

$$\det[D] = {\det[DP]}\big/{\det[P]}$$

Reduction of condition number of *D*

remove/suppress UV modes of *D*

$DP:$ Preconditioned op. IR part/IR physics

$P:$ Preconditioner UV part/UV physics

$$Z = \int \prod dU \det[D] e^{-S[U]} = \int \prod dU \det[DP] \det[P^{-1}] e^{-S[U]}$$

3. Algorithmic developments…(cont'd)

- Recent improvement strategy (HMC)

(2) Multi time step MD integrator

Multi time step MD integ. [Sexton-Weingarten, NPB 380(92)]

$$\underline{\det[DP]}\,\underline{\det[P^{-1}]} = \int d\Phi_1^{\ +} d\Phi_1 d\Phi_2^{\ +} d\Phi_2 e^{\underline{-\Phi_1^{\ +}P\Phi_1}\underline{-\Phi_2^{\ +}(DP)^{-1}\Phi_2}}$$

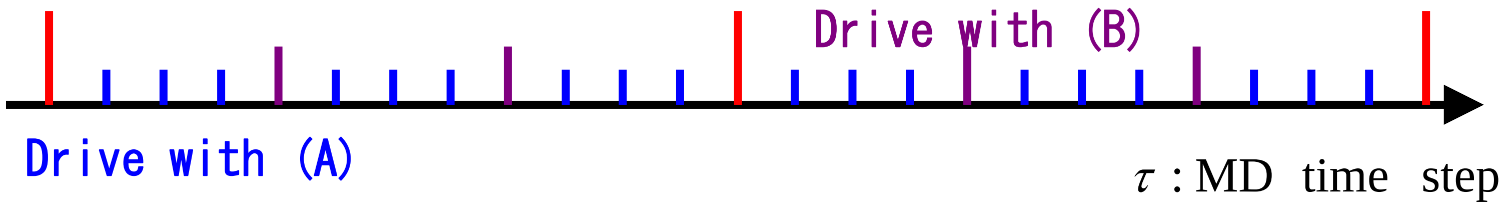
HMC partition function UV mode IR mode

$$Z = \int DUDPd\Phi_1^{\ +} d\Phi_1 d\Phi_2^{\ +} d\Phi_2 e^{-H[P,U,\Phi_1,\Phi_2]}$$

$$H = \frac{1}{2}Tr[\Pi_\mu \Pi_\mu] + \underline{S_g[U]} + \underline{S_1[U,\Phi_1]} + \underline{S_2[U,\Phi_2]}$$

$$\frac{dU_\mu}{d\tau} = i\Pi_\mu U_\mu\,, \quad \frac{d\Pi_\mu}{d\tau} = F_\mu = \boxed{F_{g\mu}} + \boxed{F_{q1\mu}} + \boxed{F_{q2\mu}} \qquad \| F_{g\mu} \| > \underline{\| F_{q1\mu} \|} > \underline{\| F_{q2\mu} \|}$$

Drive with (C)



3. Algorithmic developments…(cont'd)

(1) Transform/split det[D] using preconditioner (Action Prec.)

(A) Hasenbusch's heavy mass preconditioner

$$\det[D] = \underbrace{\det[D/D']}_{\text{IR mode}} \underbrace{\det[D']}_{\text{UV mode}}$$

D' has haevy mass than D, and $D/D' \approx 1$

(B) Geometric preconditioner (Domain Decomposition)

[Lüscher , JHEP 0305 '03,CPC 165 '05]

$$\det[D] = \det\begin{pmatrix} D_{ee} & 0 \\ 0 & D_{oo} \end{pmatrix} \det\begin{pmatrix} 1 & D_{ee}^{-1}D_{eo} \\ D_{oo}^{-1}D_{oe} & 1 \end{pmatrix} = \det[D_{ee}]\det[D_{oo}]\det[1 - D_{ee}^{-1}D_{eo}D_{oo}^{-1}D_{oe}]$$

$$= \underbrace{\det[D_{ee}]\det[D_{oo}]}_{\text{UV mode}} \underbrace{\det[\hat{D}_{ee}]}_{\text{IR mode}}$$

$\hat{D}_{ee}$: Schur complement of D

- ILU preconditing [M. Peardon, hep-lat/0011080]
- Point / stripe blocking for MG solver, Overlap kernel

[A. Boriçi, hep-lat/0704.2341; LAT2007]

3. Algorithmic developments…(cont'd)

(1) Transform/split det[D] using preconditioner (Action Prec.)

(C) *n*-th root trick and Rational approximation RHMC

[M.Clark, Ph. de Forcrand, A. Kennedy,LAT2005;
M. Clark, A.Kennedy, PRL98(2007), PRD75(2007)]

$$\det[D^\dagger D] = \det[M] = \left(\det[M^{1/n}]\right)^n \qquad Action = \sum_{j=1}^{n} \phi_j^\dagger M^{-1/n}\phi_j$$

$$\boxed{M^{1/n}}$$

UV mode suppressed

$$Action = \phi^\dagger M^{-1/n}\phi = \sum_{j=1}^{p}\phi^\dagger \frac{\alpha_j}{M+\beta_j}\phi$$

$$= \underbrace{\sum_{j=\mathrm{UVpole}}\phi^\dagger \frac{\alpha_j}{M+\beta_j}\phi}_{} + \underbrace{\sum_{j=\mathrm{IRpole}}\phi^\dagger \frac{\alpha_j}{M+\beta_j}\phi}_{}$$

Partial fraction form

$$M^{-1/n} = \sum_{j=1}^{p}\frac{\alpha_j}{M+\beta_j}$$

UV mode:large β shift.
Large MD Force, small cost

IR mode: small β shift.
Small MD Force, expensive cost

Distinctive feature: Implicit scale splitting by Rational Approx.

3. Algorithmic developments…(cont'd)

(2) MD integrator improvements

- Omelyan integrator

[Takaishi & de Forcrand,PRE73(2006); Omelyan, Mryglod & Folk,CPC151(2003)]

$$Q(\delta t)(p,q) = (p, q + \delta t \cdot p) : \text{evolve}\, q.$$

$$P(\delta t)(p,q) = (p + \delta t \cdot F, q) : \text{evolve}\, p.$$

$$\exp(t\hat{L}_H) \approx \left[Q\left(\frac{\lambda t}{n}\right) P\left(\frac{t}{2n}\right) Q\left(\frac{(1-2\lambda)t}{n}\right) P\left(\frac{t}{2n}\right) Q\left(\frac{\lambda t}{n}\right) \right]^n = \exp(t\hat{L}_{H'})$$

Shadow Hamiltonian *H'* (via Baker-Campbell-Hausdorff formula)

$$H' = H + \left(\alpha\{T,\{T,V\}\} + \beta\{V,\{T,V\}\}\right)\delta t^2 + O\left(\delta t^4\right)$$

$$\alpha = \frac{6\lambda^2 - 6\lambda + 1}{12}, \quad \beta = \frac{1 - 6\lambda}{24}$$

- Omelyan et al. minimize $\alpha^2 + \beta^2$

$$\lambda_{omelyan} = 0.19318332.....$$

Omelyan integrator / 2nd order Minimum Norm integrator (2MN)

50% improvement is observed for QCD (Takaishi & de Forcrand)

3. Algorithmic developments…(cont'd)

(2) MD integrator improvements

- Extension to Multiple time step integrator for Omelyan
  - Nesting the Kernel (QPQPQ), *K*-time scale (depth *K*)

$$\exp(t\hat{L}_H) \approx U_{K-1}(t,(n_0,n_1,\ldots,n_{K-1}))$$

$$U_j(t,(n_0,\ldots,n_j)) = \left[U_{j-1}\left(\frac{\lambda_j t}{n_j},(n_0,\ldots,n_{j-1})\right) P_j\left(\frac{t}{2n_j}\right) U_{j-1}\left(\frac{(1-2\lambda_j)t}{n_j},(n_0,\ldots,n_{j-1})\right) P_j\left(\frac{t}{2n_j}\right) U_{j-1}\left(\frac{\lambda_j t}{n_j},(n_0,\ldots,n_{j-1})\right) \right]^{n_j}$$

  - Recursively defined.

$$j = 0,1,2,\ldots,K-1,$$

$$\lambda_j : \text{ tunable parameters}$$

RBC+UKQCD, BMW, QCDSF, …

- Optimize / Customize your MD integrator
  - Shadow Hamiltonian contains errors expressed with Poisson brackets.
  - Offline measurement of Poisson brackets; exp. val. $\langle\{A,\{B,\{\ldots\}\}\}\rangle$
  - Minimize the errors by tuning integration parameter, λ , number of time scale, number of pseudo-fermions, … etc.

Takaishi & de Forcrand, PRE73 (2006);
Clark & Kennedy, LAT2007;
Poster by Kennedy

3. Algorithmic developments…(cont’d)

- Combination of the UV/IR mode separation and the Multiple time scale MD integrator is now common technique.

- There still remains the room to improve
  - UV/IR separation
    - Blocking, Rational Approx, Preconditiner ….
    - Low / IR mode : reweighting / Noisy Metropolis ....
  - MD integrator
    - Omelyan + Multiple time scale
    - Custom made MD integrator

3. Algorithmic developments…(cont'd)

Solver Improvements

(1) Mixed Precision / inner-outer solver

- Single precision : effectively doubles memory band width, data cache size, register size.
- Efficiency: S.P. > D.P. Case, mixed prec. is important.
- Intel 64/AMD 64; Single prec. > Double prec.
- Cell PS3/GPGPU; Single >> Double.

(2) Deflation Technique

- Remove / suppress small eigenvalues. Better solover behavior
- Luscher's local coherency for low modes. RG blocking like deflation.

(3) Multi Grid solver

- Adoptive Multi Grid (RG blocking) solver/preconditioner

(1) Mixed precision / Inner-Outer solver

Flexible Preconditioner

- Any iterative solver for *Ax=b.* (short recurrence solver)

  $[r \text{ and } x \text{ satisfy } r = b - Ax.]$

  $[\text{given a scalar "}\alpha\text{" and a pre-search vector "}p\text{".}]$

  $q = Ap$

  $r = r - \alpha\, q$

  $x = x + \alpha\ p$

  $[\text{new } r \text{ and } x \text{ still satisfy } r = b - Ax.]$

- Accumulated *r* and *x* should satisfy *r=b-Ax* at each update point.
- To make flexible precondition, modify the update lines as

3. Algorithmic developments…(cont'd)

(1) Mixed precision / Inner-Outer solver

- *Right* preconditioning ; *AMy = b; x = My.*

$$[r \text{ and } x \text{ satisfy } r = b - Ax.]$$

$$[\text{given a scalar "}\alpha\text{" and a pre-search vector "}p\text{".}]$$

$$\boxed{\begin{aligned} v &= Mp \\ q &= Av \end{aligned}} \; (= AMp: \quad \text{search vector for } AMy = b)$$

$$r = r - \alpha\, q$$

$$x = x + \underline{\alpha\, v}$$

$$[\text{new } r \text{ and } x \text{ still satisfy } r = b - Ax.]$$

- Search vector is computed for *AMy=b*.
- The solution-residual relation is kept for *r=b-Ax locally.*
- This enables us to change M from iteration to iteration (*Flexible* preconditioner).
- Put inner solver for $M \approx A^{-1}$
- *M* can be single precision. *r=b-Ax* is kept in double precision.

3. Algorithmic developments…(cont'd)

(1) Mixed precision / Inner-Outer solver

- CG, BiCGStab, CGS, ……, can be flexible.
  - The most simple case : Richardson / Iterative refinement. [Numerical Recipes]
    - BMW collab. uses D.P. Richardson for outer-solver + S.P. CG for inner-solver
      [BMW collab., Dürr et al.,hep-lat/0802.2706]
    - PACS-CS: uses D.P. BiCGStab+ S.P. BiCGStab

This is already common to Overlap fermions?
Low prec. sign func. (inner)
+ High prec. sign func. (outer)

- For Arnoldi type solver [GMRES,GCR…]
  - Longer reccurence relation
  - Keep a series of intermediate vectors (like *v* in prev. page.)
  - Then FGMRES, GCR(Lüscher) can be flexible.

- By tuning solver parameters
  - Most Time is spent in (inner) single precision arithmetic.
  - If the single precision kernel has much better performance than that with double precision kernel.
  - Best performance is obtained with mixed precision solver.
  - Promising for GPGPU / CELL computing!!

(2) Deflation technique

- Critical Slowing down of Solver iteration is caused by small / near zero eigenvalues.
- By subtracting such modes from the matrix spectra, we can recover from the slowding down.
- Deflation technique remove/suppress the near zero eigenspace of *D.*

This is already common to
Overlap fermions (sign functoin)

3. Algorithmic developments…(cont'd)

(2) Deflation technique （To Solve: $Ax = b \quad \cdots(1)$ ）

- Matrix A has *p*-dimensional subspace with small eigenvalues.

  Let c and u spans the subspace.

$$U_p = (u_1, u_2, \cdots, u_p)$$
$$C_p = (c_1, c_2, \cdots, c_p)$$
$$AU_p = C_p$$
$$C_p^\dagger C_p = I_p$$

- Suppose the projection operator:

$$P = I_p - C_p C_p^\dagger$$
$$Q = I_p - U_p C_p^\dagger A$$
$$PA = AQ$$

- Then consider the following preconditiond problem.

$$(PA)y = Pb \quad \cdots(2)$$

- The soluton x of Eq.(1) can be written with *y* of Eq.(2) as

$$x = Qy + U_p C_p^\dagger b$$

$$\begin{aligned} \because Ax &= AQy + AU_p C_p^\dagger b \\ &= PAy + C_p C_p^\dagger b \\ &= Pb + C_p C_p^\dagger b = b \end{aligned}$$

- Solving Eq.(2) is easier than solving Eq.(1), because the coeffcient matrix of Eq.(2) PA does not contains small eigenvalues.
- If The cost to obtain C and U is small, deflation improves solver perfomance.
- How to construct the subspace "C_p"?

3. Algorithmic developments…(cont'd)

(2) Deflation technique (cont'd)

Many works by

[Luescher, JHEP07(2007),hep-lat/0710.5417;
A.Stathopoulos, K.Orginos, hep-lat/0707.0131;
W.Wilcox, PoS(LATTICE2007),hep-lat/0710.1813;
A.Abdel-Rehim,R.B.Morgan,W.Wilcox,PoS(LATTICE2007);
R.B.Morgan,W.Wilcox,math-ph/0707.0505,math-ph/0405053;
M.L.Parks, E.De Sturler et al, SIAM J. on Sci.Comp. 28(2006)1651
LATTICE2008: Poster by Abdel-Rehim, Talk by Wilcox]

More details see Wilcox @Lat2007.

To avoid exact eigen pairs computation

(a) Overlap eigen mode computation and D^-1 computation.

- GMRES-DR,GMRES-E..:Wilcox, Morgan & Abdel-Rehim
- GCRO-DR: Parks & Sturler

These algorithms can solve *Dx=b* and eigen pairs simultaneously.

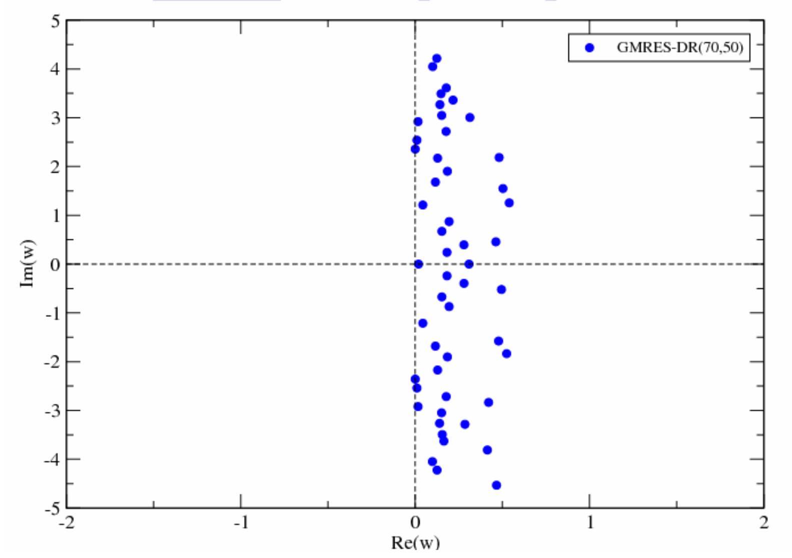
(b) Make use of Local coherency property of low modes.

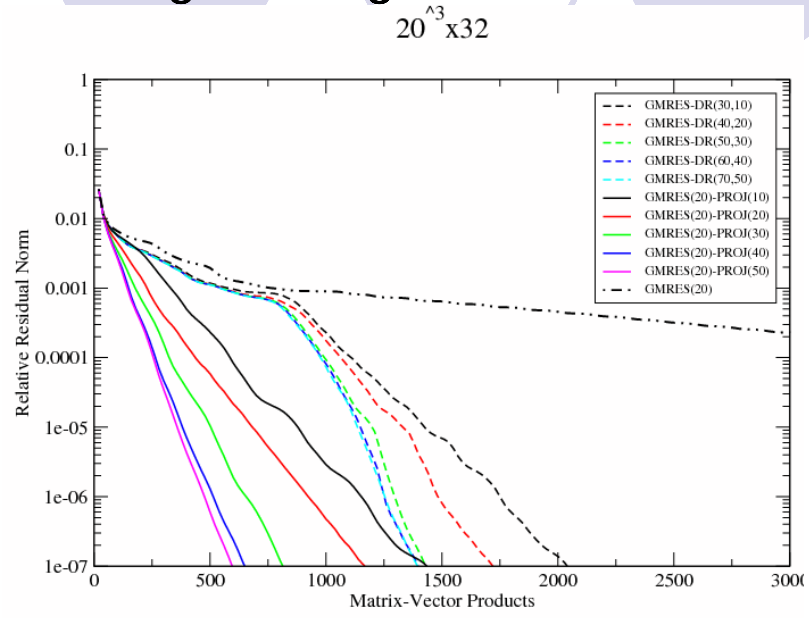
- Luscher's Domain decomposed subspace blocking with local coherency.

3. Algorithmic developments…(cont'd)

(a) Overlap eigen mode computation and D^-1 computation.

Very effective for few Near zero modes / negative eigen modes case.





[Wilcox, LAT2007]

- Near zero modes case
  - First equation or few equations are solved with GMRES-DR. Once the subspace converged, change solver with GMRES-proj, or Deflated solver.
  - Normal GMRES stagnates [dot-dot-dashed line]
  - Solver with Deflation/Projection converges. [other lines]
  - Critical slowing down is avoided.

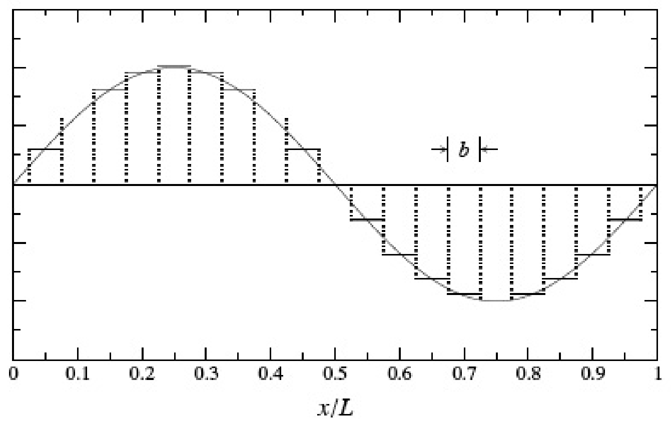
[PACS-CS collab. uses GCRO-DR for inner solver]

3. Algorithmic developments…(cont'd)

(b) Make use of Local coherency property of low modes.

Low modes can be well approximated by few blocked basis vectors [Local coherency].

[Lüscher, JHEP07(2007)081]

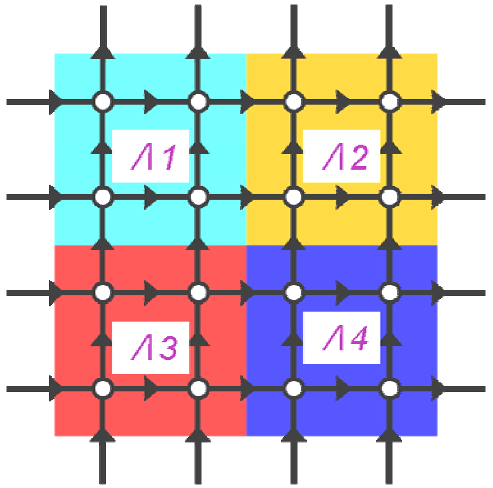


$$\text{a low mede vector}: \psi(x) \approx \sum_{\Lambda}^{blocks} \sum_{j=1}^{N} c_{(j,\Lambda)} \phi_j^{\Lambda}(x)$$

$$C = \left\{ \phi_j^{\Lambda}(x) : j = 1, \ldots, N, \Lambda = \text{all domain blocks} \right\}$$

$$\phi_j^{\Lambda}(x) = \begin{cases} \neq 0 & (x \in \Lambda) \\ = 0 & (x \notin \Lambda) \end{cases}, \quad (\phi_i^{\Lambda})^{\dagger} \cdot \phi_j^{\Lambda'} = \delta_{ij} \delta_{\Lambda\Lambda'}$$

- ϕ is constructed after few smoothing processes via inverse iteration on *N*-random vectors.
- Then blocked and orthogonalized.
  The subspace dimension is effectively enlarged:
  N x [#of Lattice blocks]
- *C*={ ϕ} spans the deflation subspace.
- Suitable for Domain-Decomposition and Memory efficient.



3. Algorithmic developments…(cont'd)

[Lüscher, JHEP07(2007)081]

(b) Make use of Local coherency property of low modes.

- Using the Low mode rich subspace *C*, the deflation projector is constructed as

$$P = 1 - DCB^{-1}C^{\dagger}, \quad Q = 1 - CB^{-1}C^{\dagger}D, \ B = C^{\dagger}DC,$$

$$PD = DQ, P^2 = P, Q^2 = Q,$$

This contains *B* which is the projection of *D* in to the subspace *C*.

- For Wilson-Dirac operator, the small Wilson-Dirac operator *B* becoms

$$B(i,\Lambda; j,\Lambda') \equiv \left\langle \phi_i^{\Lambda} \,|\, D \,|\, \phi_j^{\Lambda'} \right\rangle$$

$$= B(i, j, \Lambda)\delta_{\Lambda,\Lambda'} + \sum_{\mu=1}^{4}\left[B(i, j, \Lambda, \mu)\delta_{\Lambda+\hat{\mu},\Lambda'} + B(i, j, \Lambda, \mu)\delta_{\Lambda-\hat{\mu},\Lambda'}\right]$$

- Similar to RG blocked W.D.operator. Still has nearest neighbor interaction.
- Using this projection, critical slowing down is avoided.

3. Algorithmic developments…(cont'd)

Talk by M. Clark @ this conference.
[Brannick,Brower,Clark,Osborn,Rebbi, PRL100(2008);LAT07]

(3) MultiGrid Solver

- MultiGrid solver also removes critical slowing down.
- Choice of subspace basis is important. (Prolongator)
- Similar to Luscher's deflation. Low mode enhancement is important.

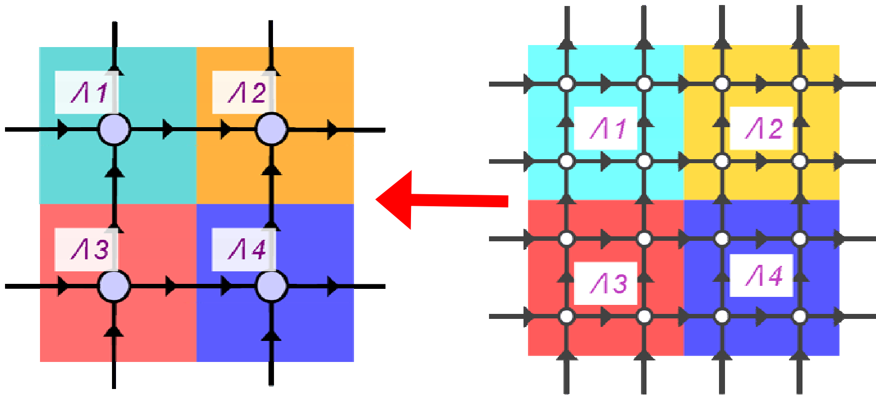
$v_l(x)$: N random vector

$$\boxed{w_l = (D)^{-k} v_l : \text{ low mode enhanced}}$$

then w_l is blocked as

$$C = \{\phi_j^\Lambda(x) : j = 1,\ldots,N, \Lambda = \text{all domain blocks}\}$$

$$\phi_j^\Lambda(x) = \begin{cases} \neq 0 & (x \in \Lambda) \\ = 0 & (x \notin \Lambda) \end{cases}, \quad (\phi_i^\Lambda)^\dagger \cdot \phi_j^{\Lambda'} = \delta_{ij}\delta_{\Lambda\Lambda'}$$



- To solve $Dx = b$, use the preconditione defined by

$$P \equiv CB^{-1}C^\dagger \quad \text{with} \quad B \equiv C^\dagger DC, \text{ or } \quad B(i,\Lambda; j,\Lambda') \equiv \langle \phi_i^\Lambda | D | \phi_j^{\Lambda'} \rangle$$

3. Algorithmic developments…(cont'd)

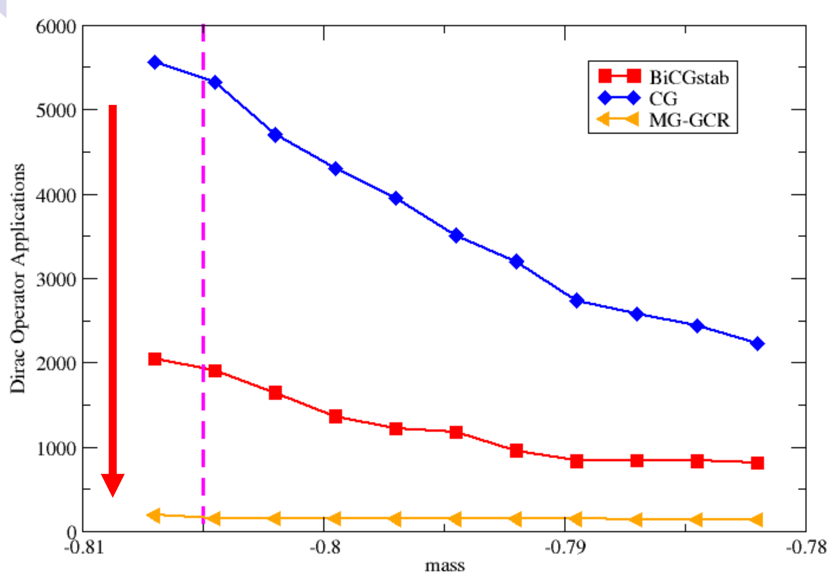
(3) MultiGrid Solver (cont'd)

- Then Solve

$$PDx = Pb$$

- *P* is the approximation of *D^-1* in the subspace *C*.
- *P* contains *B*^-1. to solve this next blocking is applicable.
- Recursively applying this blocking.
  ⇒ MultiGrid. *V* cycle
- Similar to Luscher's deflation subspace blocking. Low mode enhancement is important.

No critical slowing down



Mass

QCD 16^3x32 Wilson Case
Talk by M. Clark @ this conference.
[Brannick,Brower,Clark,Osborn,Rebbi, PRL100(2008);LAT07]

Another RG blocking by A. Borici, hep-lat/0704.2341; LAT2007.

3. Algorithmic developments…(cont'd)

Solver Works in this conference:
- J.Bloch (for Overlap fermion) [Mon. Chesapeake C],
- J.Osbon (Initial guess for multi-shift solv.)[Mon. Chesapeake C]
- W.Wilcox (Deflation/Lanczos/multiple)[Mon. Chesapeake C]
- A.Abdel-Rehim (Seed method/multiple) [Poster S.A]

- Mixed precision solver effectively enhances the solver performance.
  - application to GPGPU/CELL?
- Deflation and MultiGrid blocking with low mode-rich basis vector removes Critical slowing down.

3. Algorithmic developments…(cont'd)

Algorithm works in this conference:

[July 15, Tue. Chesapeake B]

●A. Bazavov (for HISQ action dynamical sim.)
●R.C. Brower (Mobius Algorithm for DW/GapDW fermion.)
●M. Clark (Remove Critical Slowing down)
●T. Kruth (Dynamically Smeared Fermions)

[July 16, Wed. Chesapeake B]

●O. Witzel (Polynomial HMC)
●R. Renfrew (Reduce Ch.Sym.breaking for DW)
●F. Palombi (Reweighting for Low mode Quark determinant)
●W. Cherrington (Dual Lattice Algorithm)
●J. Mucci (SiCortex Machines)

[July 15, Tue. Poster session]

●A. Pochinsky (Efficient QCD code made simpler: qa0)
●L. Piccoli (Tracking QCD workflows)
●G. von Hippel (Petrurbative imp. with HISQ fermions)

4. Outlook: Physics at 1PFlops

- Dynamical QCD simulation at 1 PFlops
  - Physical quark masses (*Mud* < 10 MeV, L=3fm,a=0.1fm)
    - Cost O(10) Tflops Years Wilson/KS type

      [ALPHA,BMW,CERN,ETM,JLAB,PACS-CS,QCDSF,MILC,..]
    - O(100) Tflops Years? Overlap/DW type

      [UKQCD/RBC,JLQCD/TWQCD,SESAM/QCDSF,…]
  - Finer lattice spacing (1/*a* > 6 GeV?, L=2fm, 64^3x128 lattice)
    - Charm quarks

$$\Lambda_{QCD} \approx 0.3\mathrm{GeV} < m_{charm} \approx 1.5\mathrm{GeV} < 1/a \approx 6\mathrm{GeV}$$

$$1/\Lambda_{QCD} \approx 0.6\mathrm{fm} > 1/m_{charm} \approx 0.13\mathrm{fm} > a \approx 0.03\mathrm{fm}$$

    - Continuum limit

$$(am_c)^2 \approx (1.5/6)^2 = 0.06 \quad (6\% \text{ error})$$

  - Larger lattice volume (*L* > 6 fm?, 1/a=2GeV, 64^3x128 lattice)
    - Multi hadron system

$$m_\pi \approx 0.1\mathrm{GeV} < \Lambda_{QCD} \approx 0.3\mathrm{GeV} < 1/a \approx 2\mathrm{GeV}$$

$$1/m_\pi \approx 2\mathrm{fm} > 1/\Lambda_{QCD} \approx 0.6\mathrm{fm} > a \approx 0.1\mathrm{fm}$$

Multi scale physics

4. Outlook: Physics at 1PFlops

- Dynamical QCD simulation at 1 PFlops
  - Empirical cost formula

[Wilson/KS type]

$$\mathrm{Cost[TFlopsYears]} = C\left[\frac{\#\mathrm{Conf}}{100}\right]\cdot\left[\frac{20\mathrm{MeV}}{\overline{m}_q}\right]^3\cdot\left[\frac{L}{3\mathrm{fm}}\right]^5\cdot\left[\frac{0.1\mathrm{fm}}{\mathrm{a}}\right]^7$$

[Ukawa,Lat2001@Berlin]

$$\mathrm{Cost[TFlopsYears]} = K\left[\frac{\#\mathrm{Conf}}{100}\right]\cdot\left[\frac{20\mathrm{MeV}}{\overline{m}_q}\right]^1\cdot\left[\frac{L}{3\mathrm{fm}}\right]^5\cdot\left[\frac{0.1\mathrm{fm}}{\mathrm{a}}\right]^6$$

[DDHMC: Del Debbio et al..JHEP0702(2007)056]

- Now O(10)TFlopsYears for $M_\pi / M_\rho \approx 0.2$ at $a \approx 0.1\mathrm{fm}, L \approx 3\mathrm{fm}$

[Talk by Kuramashi, PACS-CS]

- Finer lattice spacing (1/*a* > 6 GeV?, L=2fm, 64^3x128 lattice)
  - Charm quark on fine lattcie requires a=0.03 fm lattice.

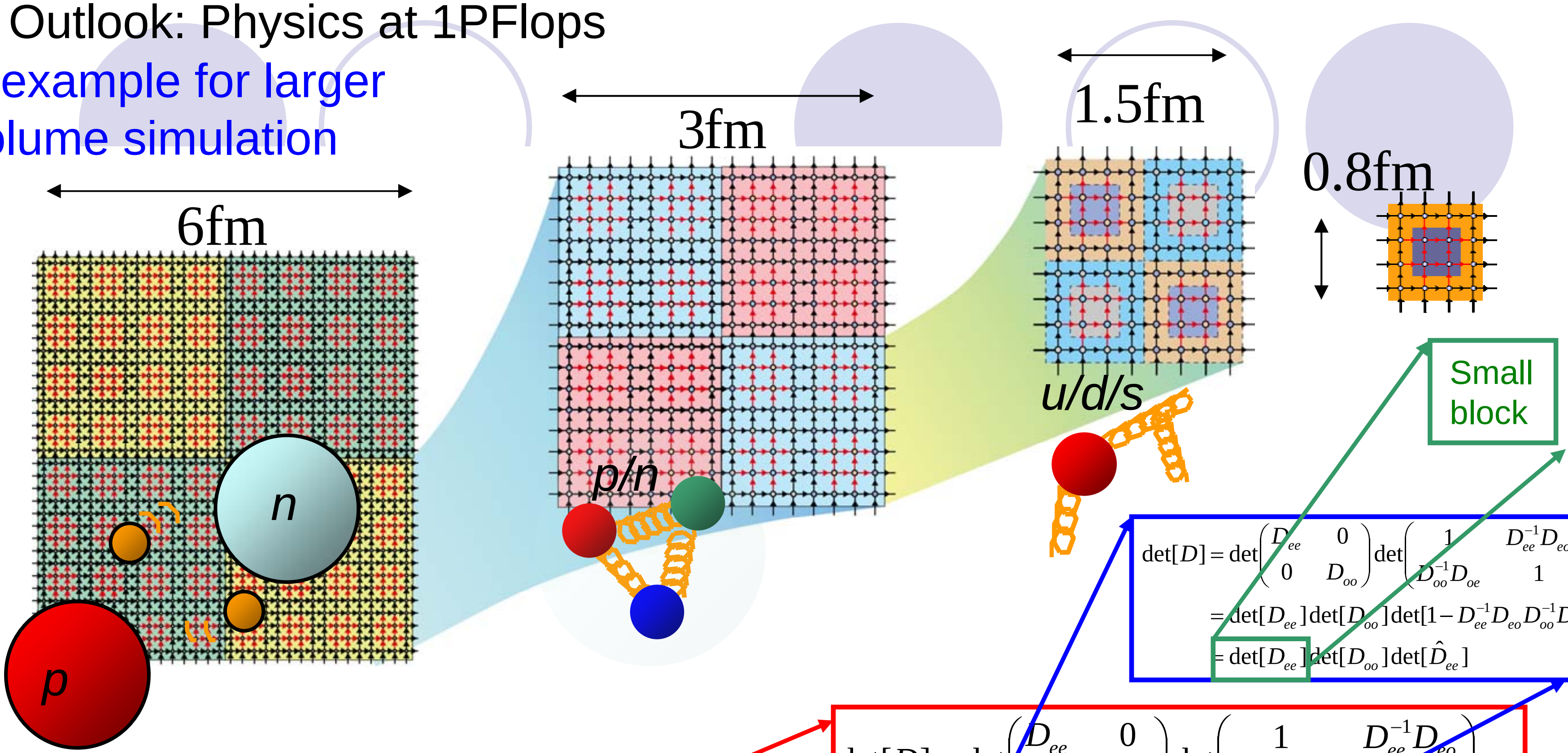
    The Cost is (2/3)^5*3^6 =96 larger. ⇒ O(1) Pflops Years is required.
  - Still difficult problem? ⇒ 10PFlops probrem.
- Larger lattice volume (*L* > 6 fm?, 1/a=2GeV, 64^3x128 lattice)
  - Multi hadron system by doubling the lattice extent.

    The Cost is 2^5 =32 larger. ⇒ O(300) Tflops Years is required.

30% sustained speed with 1 PFlops peak speed machine can handle this problem.

4. Outlook: Physics at 1PFlops

A example for larger volume simulation



$$\det[D] = \det\begin{pmatrix} D_{ee} & 0 \\ 0 & D_{oo} \end{pmatrix} \det\begin{pmatrix} 1 & D_{ee}^{-1}D_{eo} \\ D_{oo}^{-1}D_{oe} & 1 \end{pmatrix}$$

$$= \det[D_{ee}]\det[D_{oo}]\det[1 - D_{ee}^{-1}D_{eo}D_{oo}^{-1}D_{oe}]$$

$$= \boxed{\det[D_{ee}]}\det[D_{oo}]\det[\hat{D}_{ee}]$$

$$\det[D] = \det\begin{pmatrix} D_{ee} & 0 \\ 0 & D_{oo} \end{pmatrix} \det\begin{pmatrix} 1 & D_{ee}^{-1}D_{eo} \\ D_{oo}^{-1}D_{oe} & 1 \end{pmatrix}$$

$$= \det[D_{ee}]\det[D_{oo}]\det[1 - D_{ee}^{-1}D_{eo}D_{oo}^{-1}D_{oe}]$$

$$= \boxed{\det[D_{ee}]}\det[D_{oo}]\det[\hat{D}_{ee}]$$

$$\det[D] = \det\begin{pmatrix} D_{ee} & 0 \\ 0 & D_{oo} \end{pmatrix} \det\begin{pmatrix} 1 & D_{ee}^{-1}D_{eo} \\ D_{oo}^{-1}D_{oe} & 1 \end{pmatrix}$$

$$= \det[D_{ee}]\det[D_{oo}]\det[1 - D_{ee}^{-1}D_{eo}D_{oo}^{-1}D_{oe}]$$

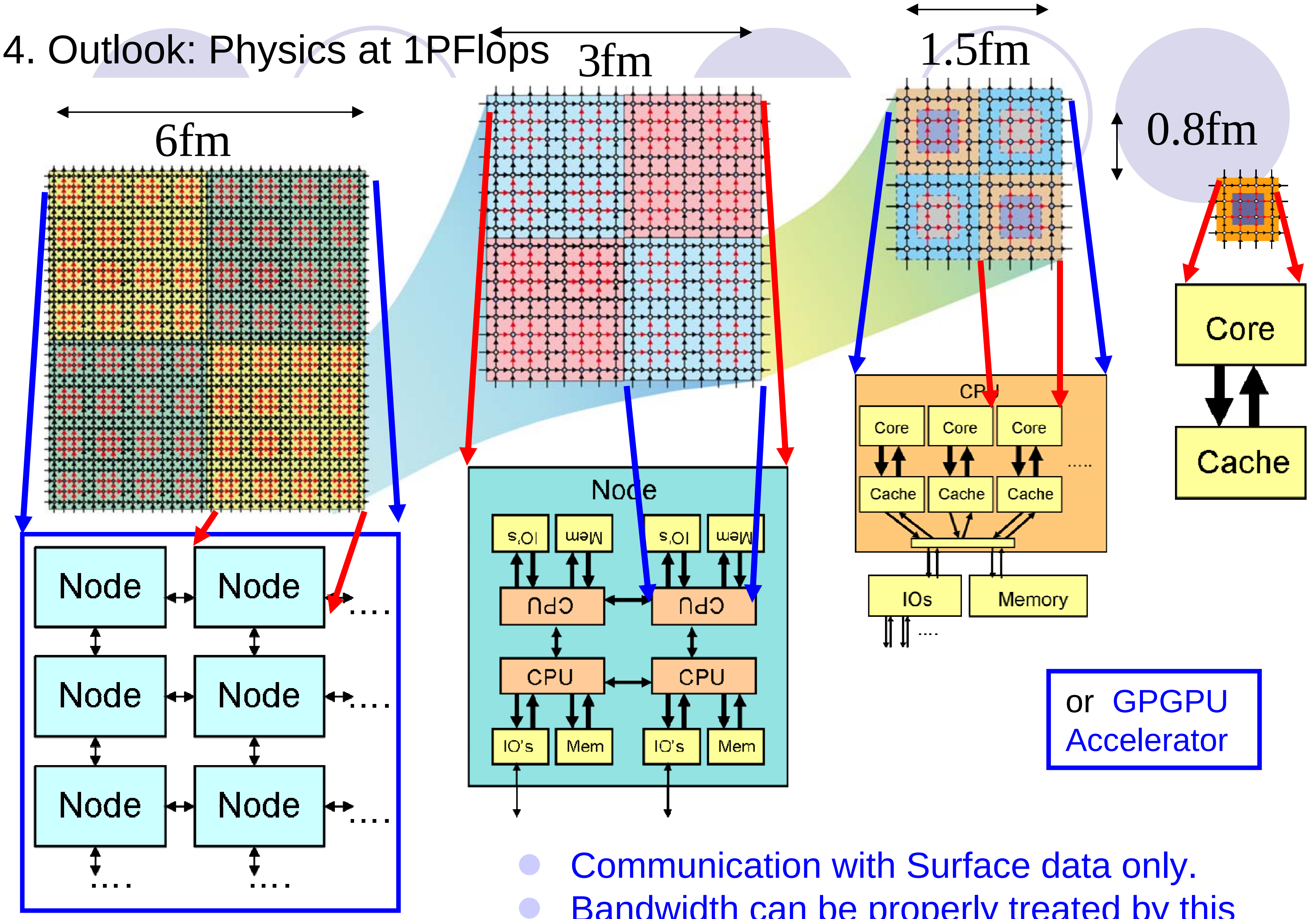
$$= \boxed{\det[D_{ee}]}\det[D_{oo}]\det[\hat{D}_{ee}]$$

(L > 6 fm?, 1/a=2GeV, 64^3x128 lattice)

- Nested Domain Decomposition +Some Improvement technology.

[Lüscher, JHEP 0305 (2003) 052]

4. Outlook: Physics at 1PFlops



or GPGPU Accelerator

- Communication with Surface data only.
- Bandwidth can be properly treated by this blocking.
- But Latency is limited by speed of light.

5. Summary

- Machine trends
  - Multi core architecture is the trend.
  - GPGPU has better cost performance, but actual application for LQCD is now beginning. Large scale simulation is still missing.
  - CELL becomes common archtecture for HPC?
- Algorithm
  - UV/IR separation + multiple time step MD is common.
  - Deflation and MG remove critical slowing down.
- Physics at 1 PFlops
  - Large volume simulation for multi hadron system can be a target. [Multi scale physics]
  - To tread Multi scale physics, the structure of machine architecture should be taken account.

That's all Thank you!

Backup slides

4. Outlook: Physics at 1PFlops

- Wilson/KS type fermion can handle multi-hadron system with 1Pflops machine in principle.
- Whole System performance analysis that has been done, for ex. QCDOC, CP-PACS, APE…., is *again* required.

- Domain-Wall / Overlap fermion : Are there this kind of decomposition ?
- D.W. / 5D-rep. Overlap can use geometric preconditoner.
- 4D-Overlap requires special kernel for geometric decomposition?

  Dirichlet boundary condition for OV op. [Luscher, "Shrodinger Functional with exact Chiral symmetory", JHEP 0605 (2006) 042]

  - Enormous works for Dynamical Overlap/DW fermions

    [Many people ,RBC,QCDSF,SESAM,JLQCD,……..]

- QCD Software / infrastructure works

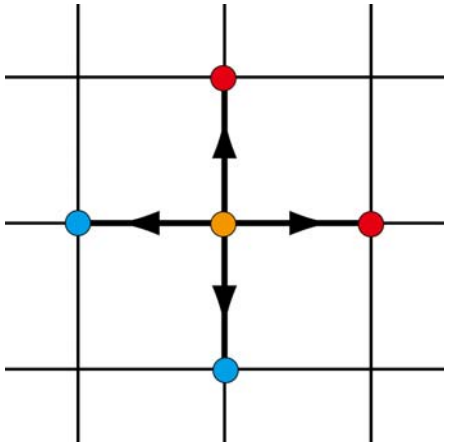
  [MILC code; ILDG; B.Joo,USQCD; A.Borici,QCDLAB; …]

That's all Thank you!

2. Machine Trends (cont'd)

- For QCD (dynamical)
  - Hybrid Monte Carlo (HMC)
  - Dynamical Quark part requires huge amount of hopping matrix multiplication.

$$M(n,m) = \sum_{\mu=1}^{4} \left[(1-\gamma_\mu) U_\mu(n) \delta_{n+\hat{\mu},m} + (1+\gamma_\mu) U_\mu^\dagger(m) \delta_{n-\hat{\mu},m} \right]$$



  - This computation requires

$\sim$ 3 Byte/Flop for a site…

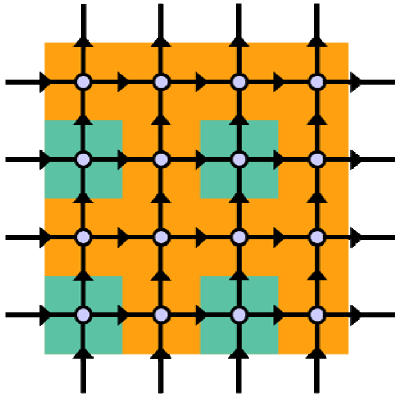
Register, Cache, are memory blocking are required at each layer.

3. Algorithmic developments…(cont'd)

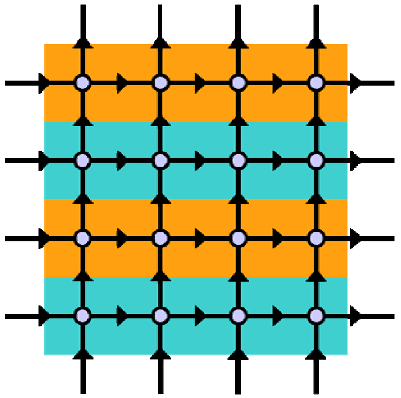
(1) Transform/split det[D] using preconditioner (Action Prec.)

(b') Point / stripe (RG) blocking for MG solver, Overlap kernel

[A. Borici, hep-lat/0704.2341; LAT2007]



type1



type2

Change Site Ordering

$$\det[D] = \det\begin{pmatrix} D_{bb} & D_{br} \\ D_{rb} & D_{rr} \end{pmatrix}$$
$$= \det[D_{rr}]\det[D_{bb} - D_{br}D_{rr}^{-1}D_{rb}]$$
$$= \det[D_{rr}]\det[S_{bb}]$$

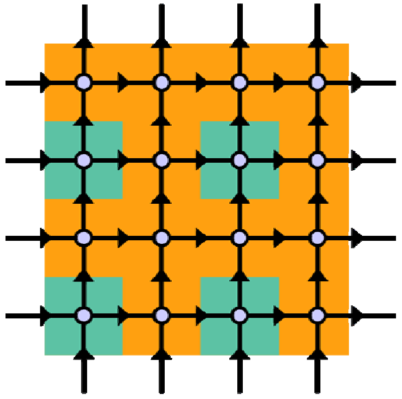
UV mode IR mode

S_{bb} : Schur complement of D

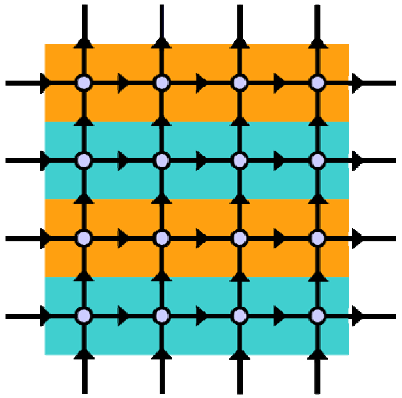
3. Algorithmic developments…(cont'd)

(b') Point / stripe (RG) blocking for MG solver, Overlap kernel

[A.Borici, hep-lat/0704.2341; LAT2007]

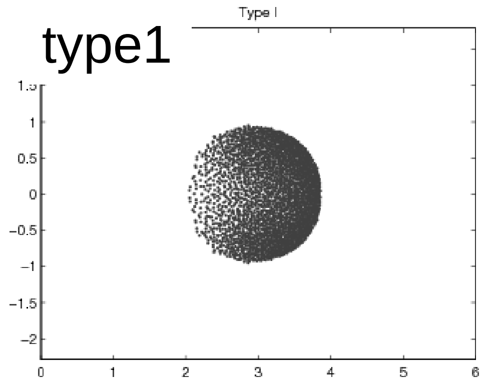


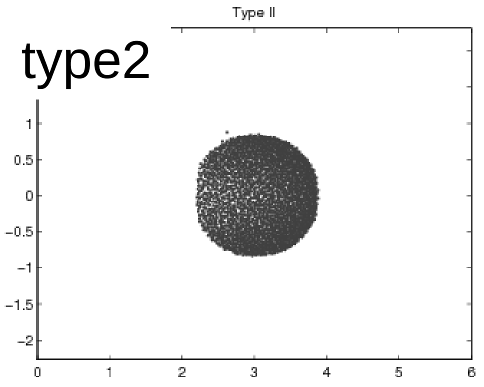
type1

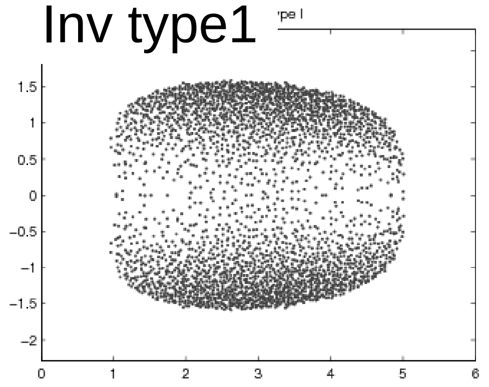


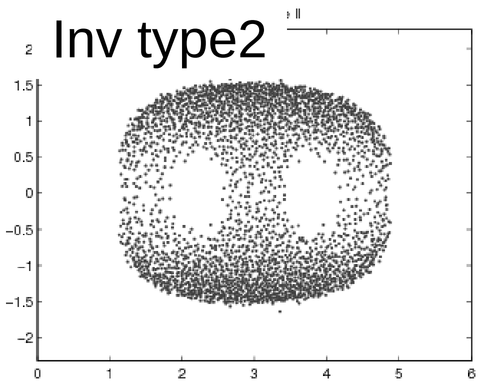
type2

S_{bb}









β =5.4, 8^4 lattice

3. Algorithmic developments…(cont'd)

(c) *n*-th root trick and Rational approximation RHMC

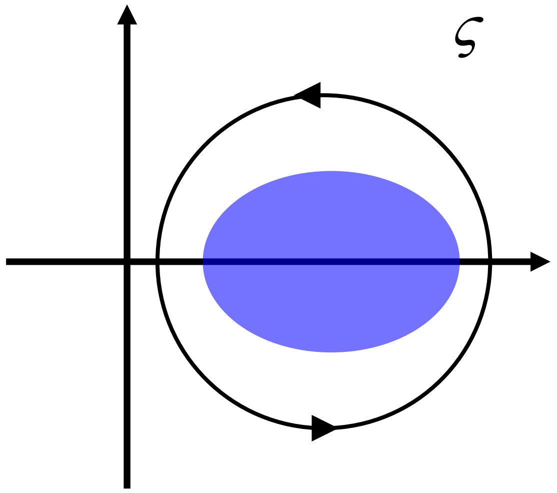
- Rational approximation
  - C.f. Multi boson algorithm $1/x \approx \sum_j c_j x^j$
    - Hermitian Polynomial approx. (Luscher '93)
    - Non-Hermitian Polynomial approx. (Borrelli, de Forcrand, Galli '96)

For RHMC algorithm , similar variant is possible.

- Hermitian Rational approx. VS Non-Hermitian Rational approx.

$$D^{-1/n} = \frac{1}{2\pi i}\oint \varsigma^{-1/n}(\varsigma - D)^{-1} d\varsigma$$
$$\approx \sum_j (\varsigma_j)^{-1/n}(\varsigma_j - D)^{-1} d\varsigma_j$$

D : Non - Hermitian Wilson Dirac op.



(2) MD integrator improvements

- Omelyan integrator
- Simple leapfrog

[Takaishi & de Forcrand,PRE73(2006);
Omelyan, Mryglod & Folk,CPC151(2003)]

$$H(p,q) = T(p) + V(q), \quad : \text{Hamiltonian}$$

$$\hat{L}_H X \equiv \{X, H\} = \frac{\partial X}{\partial q}\frac{\partial H}{\partial p} - \frac{\partial H}{\partial q}\frac{\partial X}{\partial p}$$

$$\begin{pmatrix} p(t) \\ q(t) \end{pmatrix} = \exp(t\hat{L}_H)\begin{pmatrix} p(0) \\ q(0) \end{pmatrix}$$

$$Q(t) \equiv \exp(t\hat{L}_T) : \text{evolve } q.$$

$$P(t) \equiv \exp(t\hat{L}_V) : \text{evolve } p.$$

$$\exp(t\hat{L}_H) \approx \left[Q\left(\frac{t}{2n}\right)P\left(\frac{t}{n}\right)Q\left(\frac{t}{2n}\right)\right]^n$$

Leapfrog integrator

- This operator does not conserve H, but conserves Shadow Hamiltonian H'.

$$\left[Q\left(\frac{t}{2n}\right)P\left(\frac{t}{n}\right)Q\left(\frac{t}{2n}\right)\right]^n = \exp(t\underline{\hat{L}_{H'}})$$

Exact rel.

Shadow Hamiltonian: $H' = H - \frac{1}{24}\left(\{T,\{T,V\}\} + 2\{V,\{T,V\}\}\right)\delta t^2 + O\left(\delta t^4\right)$

(2) Deflation technique

- LQCD requires thousand of linear equation solution

$$Dx^{(i)} = b^{(i)}, i = 1,2,3\ldots$$

$$D^{(i)}x^{(i)} = b^{(i)}, i = 1,2,3\ldots, D^{(i)} \approx D^{(i-1)},$$

  - Multiple right-hand side or chain of linear equations.
    - Quark propagator
    - Solver in HMC trajectory
- The reduction of condition number of coefficient matrix *D* is very effective.   Efficient Preconditioning is desired.
- Deflation technique is one of the efficient technique to reduce the condition number.
  - Deflation remove/suppress small eigenspace of *D.*

3. Algorithmic developments…(cont'd)

(2) Deflation technique (cont'd)

(a) Overlap eigen mode computation and D^-1 computation.

Use Arnoldi type Solver [Krylov subspace method] for *Ax*=*b*.

$$r = b - Ax_0 : \text{ initial residual}$$

$$v_0 = r/|r|,$$

$$AV_K = V_{K+1}\overline{H}_K : \text{ Arnoldi factorization (via Gram-Schmidt).}$$

$$V_{K+1} = (v_0, v_1, \ldots, v_{K+1}), \quad v_i^\dagger v_j = \delta_{ij}, \quad \text{Krylov subspace basis.}$$

$$\overline{H}_K : \text{ upper Hessenberg matrix } (K+1 \times K)$$

$$x = x_0 + V_{K+1}c, \quad \text{where } c^T = (c_0, c_1, \ldots, c_{K+1})$$

$$\text{Minimize: } |r| = |b - Ax| \text{ with } c.$$

GMRES(K)

- V_{K+1} and H_K contains the spectrum info. of *A*.
- At restarting, construct Harmonic-Ritz pairs.

$$\text{Solve small eigen problem } (K \times K)$$

$$(\overline{H}_K^\dagger \overline{H}_K - \mu H_K^\dagger) y = 0$$

GMRES-DR/GCRO-DR

$$\text{Harmonic Ritz pair } (\mu, w = V_K y) \text{ is approxmation for } (A - \mu)w = 0.$$

$$\text{Few } \{w\} \text{ basis vectors are recycled as deflation subspace for the next iteration.}$$

Reduce eigen mode comp. cost

3. Algorithmic developments…(cont'd)

[Lüscher, JHEP07(2007)081]

(b) Make use of Local coherency property of low modes.

- Deflation projector contains small linear equation *B^-1*.

$$P = 1 - DCB^{-1}C^{\dagger}, \quad Q = 1 - CB^{-1}C^{\dagger}D, \ B = C^{\dagger}DC,$$

$$PD = DQ, P^2 = P, Q^2 = Q,$$

- For Wilson-Dirac operator, the small Wilson-Dirac operator *B* becoms

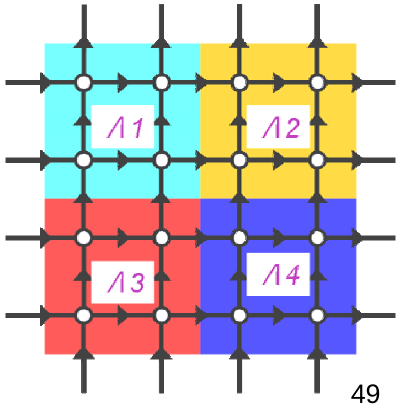
$$B(i,\Lambda; j,\Lambda') \equiv (\phi_i^{\Lambda})^{\dagger}\cdot(D\phi_j^{\Lambda'})$$

$$= B(i,j,\Lambda)\delta_{\Lambda,\Lambda'} + \sum_{\mu=1}^{4}\left[B(i,j,\Lambda,\mu)\delta_{\Lambda+\hat{\mu},\Lambda'} + B(i,j,\Lambda,\mu)\delta_{\Lambda-\hat{\mu},\Lambda'}\right]$$

- Similar to RG blocked W.D.operator. Still has nearest neighbor interaction.
- To avoid frequent application of Projection, *P* is applied to SAP preconditioned problem:

$$DM_{SAP}y = b, x = M_{SAP}y,$$

$$\Rightarrow \ PDM_{SAP}y = Pb, \ x = QM_{SAP}y + CB^{-1}C^{\dagger}b.$$

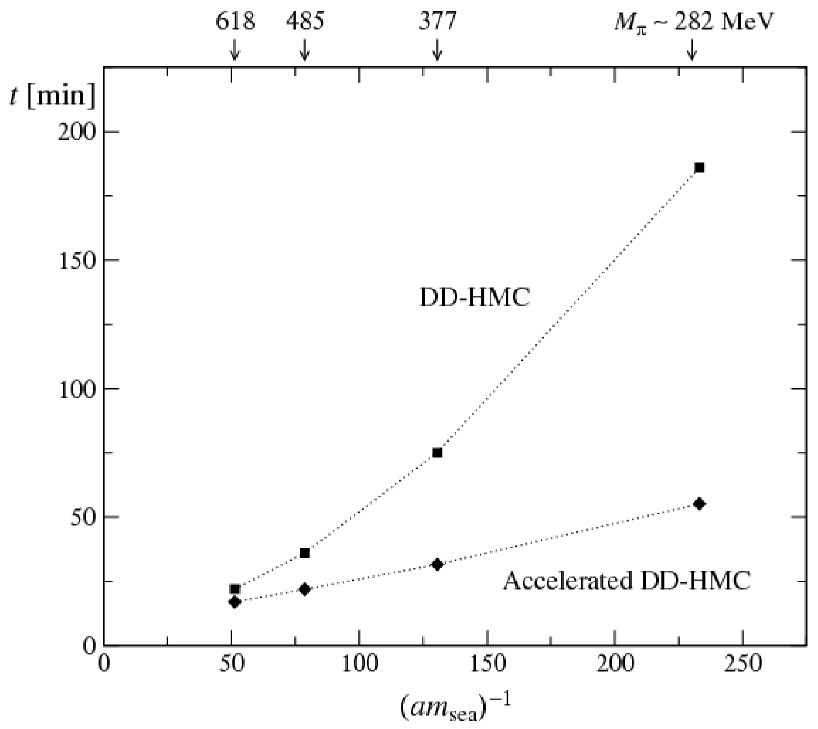


3. Algorithmic developments…(cont'd)

[Lüscher,hep-lat/0710.5417]

(b) Make use of Local coherency property of low modes.

- Deflation accelerates DDHMC performance
- Factor 2-3 improvement is observed.
- Speedup is significant for smaller quark masses.



Deflation removes critical slowing down.

3. Algorithmic developments…(cont'd)

(c) *n*-th root trick and Rational approximation RHMC

Further cost reduction using Rational approximation

$$M^{-1/n} = \sum_{j=1}^{p} \frac{\alpha_j}{M+\beta_j}$$

M: Hermitian, spectrum boundary is known.

α , β : real parameter

Optimal Chebyshev approx.

$$Action = \phi^\dagger M^{-1/n}\phi = \sum_{j=1}^{p} \phi^\dagger \frac{\alpha_j}{M+\beta_j}\phi$$

Partial fraction form

$$= \underbrace{\sum_{j=\mathrm{UVpole}} \phi^\dagger \frac{\alpha_j}{M+\beta_j}\phi}_{} + \underbrace{\sum_{j=\mathrm{IRpole}} \phi^\dagger \frac{\alpha_j}{M+\beta_j}\phi}_{}$$

UV mode:large β shift. IR mode: small β shift.

RHMC: RBC+UKQCD, DW Nf=2+1 simulation [hep-lat/0804.0473;PRD76(2007)]
Clark and Kennedy KS fermion [hep-lat/0610047;PRD75(2007)]
Takaishi and Nakamura, One-flavor Wilson fermion F.T. [LAT2007,hep-lat/0711.3888]

3. Algorithmic developments for dynamical QCD
(Wilson type)

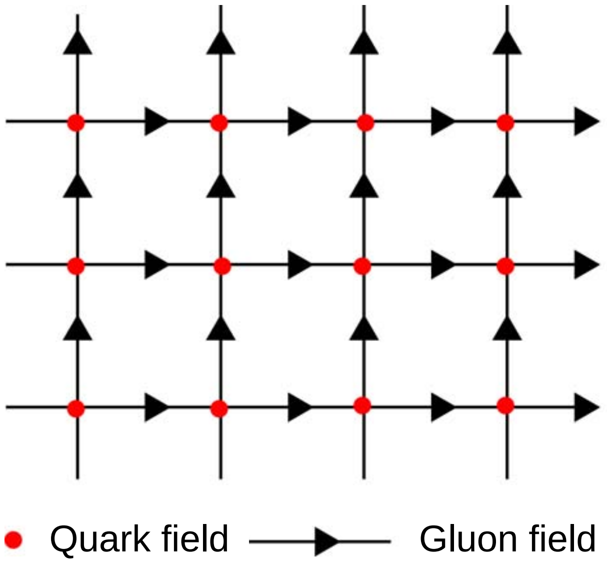
- Lattice QCD partition function

$$Z = \int \prod dU d\psi e^{-S[U,\psi]}$$

$S[U,\psi] = S_G[U] + S_Q[U,\psi]$: Lattice QCD action

$S_G[U]$: Gluon part $(\to Tr[F_{\mu\upsilon}F_{\mu\upsilon}]/(4g^2))$

$S_Q[U,\psi]$: Quark part $(\to \sum_f \overline{\psi}_f (D+m_f)\psi_f)$



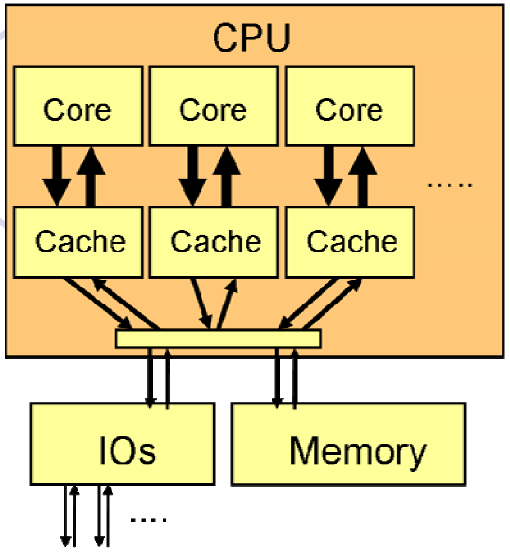
$\psi(n)$ $U_\mu(n)$

- Nf=2+1 partition function
  (ψ integ.out)

$$Z = \int \prod dU \det[D_{ud}]^2 \det[D_s] e^{-S[U]}$$

HMC algorithm to generate {*U*}.

2. Machine Trends (cont'd)



- 1～10 PFlops machine Bottlenecks
  - Memory band width
    - DDR3(1333) 10GB/sec
    - RambusXDR 26GB/sec
    - Byte/Flop < 0.25 (single CPU)
    - GPGPU is more better 100GB/sec

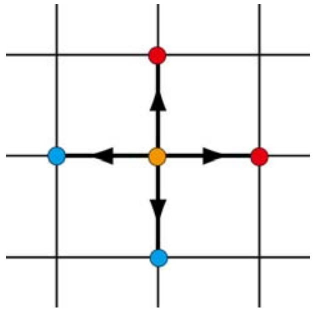
  Multi slots/node enhances the node speed. [SMP or NUMA] but…

  - IO/ Network band width

    Depends on the NIC but
    - Myrinet 10G 1.25GB/sec
    - Infiniband DDR 2.0GB/sec
    - Ex. Byte/Flop < 2/48 = 0.04
    - To balance, multi rail (x4 or x8…)

$$M(n,m) = \sum_{\mu=1}^{4}\left[(1-\gamma_\mu)U_\mu(n)\delta_{n+\hat{\mu},m} + (1+\gamma_\mu)U_\mu^\dagger(m)\delta_{n-\hat{\mu},m}\right]$$

Hopping Mult :
～3Byte/Flop req.



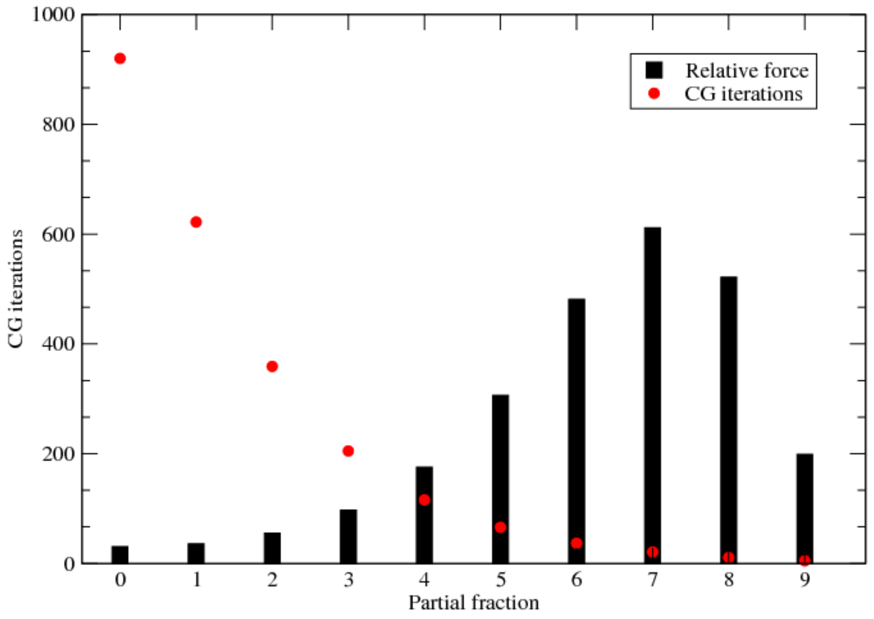
Blocking is required at each level (core/cpu/node) for 1PFlops machine

3. Algorithmic developments…(cont'd)

(c) *n*-th root trick and Rational approximation RHMC

Further cost reduction using Rational approximation

$$Action = \phi^\dagger M^{-1/n}\phi = \sum_{j=\mathrm{UVpole}} \phi^\dagger \frac{\alpha_j}{M+\beta_j}\phi + \sum_{j=\mathrm{IRpole}} \phi^\dagger \frac{\alpha_j}{M+\beta_j}\phi$$



DW β =2.13,
24^3x64x16
m_{ud}/m_s=0.25
RHMC force norm

IR mode: small β shift
Expensive cost, small force
=> Coarser MD step

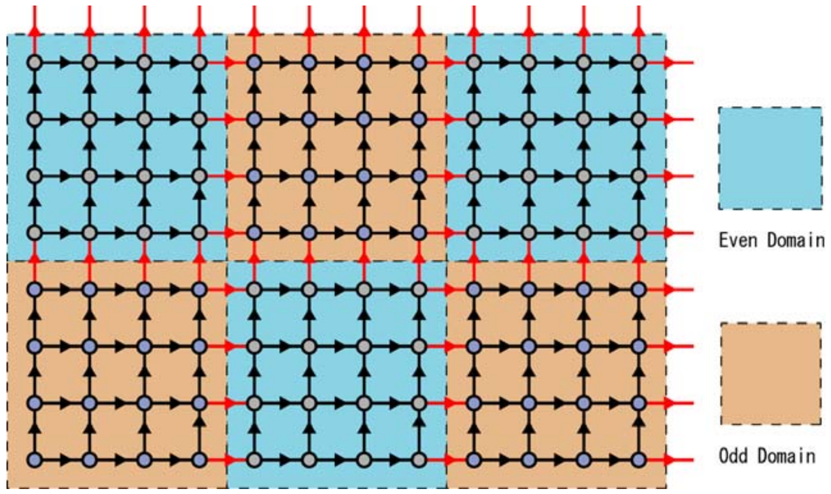
UV mode:large β shift
Cheap cost, large force
=> finer MD step

3. Algorithmic developments…(cont'd)

(1) Transform/split det[D] using preconditioner (Action Prec.)

(b) Luscher Domain-Decomposition preconditioned DDHMC

[Lüscher , JHEP 0305 '03,CPC 165 '05]



$$\det[D] = \det\begin{pmatrix} D_{ee} & 0 \\ 0 & D_{oo} \end{pmatrix} \det\begin{pmatrix} 1 & D_{ee}^{-1}D_{eo} \\ D_{oo}^{-1}D_{oe} & 1 \end{pmatrix}$$

$$= \det[D_{ee}]\det[D_{oo}]\det[1 - D_{ee}^{-1}D_{eo}D_{oo}^{-1}D_{oe}]$$

$$= \underbrace{\det[D_{ee}]\det[D_{oo}]}_{\text{UV mode}}\underbrace{\det[\hat{D}_{ee}]}_{\text{IR mode}}$$

DDHMC simulations:
- ●ALPHA: Von Hippel
- ●CERN: Luscher, Debbio, Giusti, Petronzio
- ●PACS-CS

$\hat{D}_{ee}$: Schur complement of D

How about another decomposition/blocking?

- ILU preconditing [M. Peardon, hep-lat/0011080]
- Point / stripe blocking for MG solver, Overlap kernel

[A. Boriçi, hep-lat/0704.2341; LAT2007]

(2) MD integrator improvements

- Optimize / Customize your MD integrator

Takaishi & de Forcrand, PRE73 (2006);
Clark & Kennedy, LAT2007;
Poster by Kennedy

- Shadow Hamiltonian contains errors expressed with Poisson brackets.
- Offline measurement of Poisson brackets;
  exp. val. $\langle\{A,\{B,\{\ldots.\}\}\}\rangle$
- Minimize the errors by tuning integration parameter, λ , number of time scale, number of pseudo-fermions, … etc.