X-Ability
クロスアビリティ

GPGPUによるフラグメント分子軌道法の高速化

/* フラグメント分子軌道法(FMO)およびエネルギー表示法(ER)を活用した自由エネルギー計算のGPGPUによる高速化 */

Acceleration of fragment molecular orbital method by GPGPU

Ryota Koga　古賀 良太

President of X-Ability Co.,Ltd. ㈱クロスアビリティ

Collaborative Researchers

Yuki Furukawa(X-Ability Co.,Ltd.), Koji Yasuda(Nagoya University)

Nobuyuki Matsubayashi, Shun Sakuraba(Kyoto University)

30 May 2012

第17回計算工学講演会＠京都教育文化センター

X-Ability
クロスアビリティ

X-Ability Co.,Ltd. ㈱クロスアビリティ

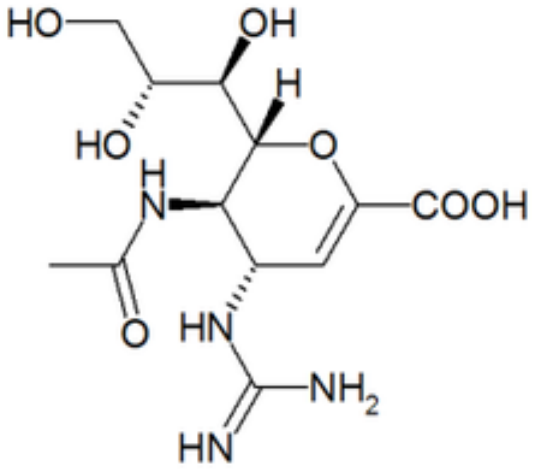
- 5 Office (foundation : 15th Jan 2008)
  Japan : Hongo, Tokyo + The K-comp, Kobe
  Thailand, Indonesia, and U.S. branch

- 2 Business
  (1) Scientific Computing (mainly Chemistry)
  GPGPU computation
  (2) Sensor Networking

- By only 3 Regular Board Members

X-Ability
クロスアビリティ

Drugs using Computational Chemistry

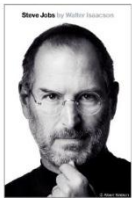


©wikipedia
Zanamivir
trade name **Relenza**

Influenza preventive medicine

©wikipedia
Erlotinib hydrochloride
(trade name **Tarceva**)

Lung cancer, pancreatic cancer

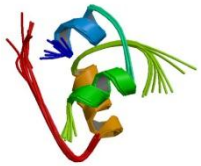


***Steve Jobs* might have used this to extend his life…**

X-Ability
クロスアビリティ

Energy Representation (ER)

- Free energy computation method which is of comparable accuracy to exact method but is faster by two orders of magnitude
- Ermod http://sourceforge.net/p/ermod/wiki/Home/
  - Product at a Japanese national project
  - Applicable to supercritical fluid, ionic liquid, micelle, and lipid membrane
  - All-atom computation of free energy of protein solvation is now feasible.



  - **Current theoretical challenge is the ligand-binding into protein.**

X-Ability
クロスアビリティ

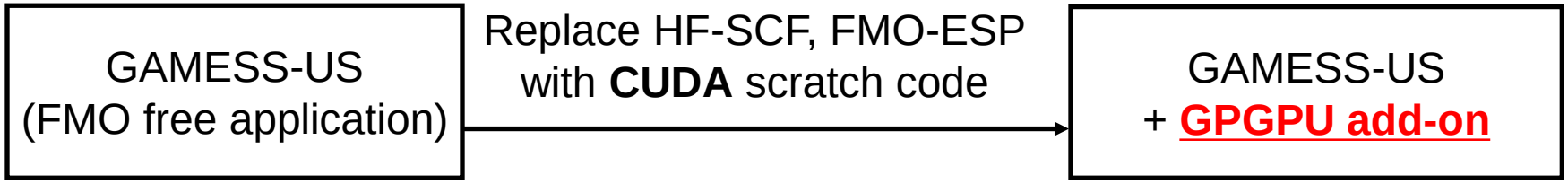
Trial of free energy calc on GPGPU

- **(a) FMO based MD # time consuming**
  - MD using forcefield with QM(FMO) charge
- **(b) MD/ER**
  - Free energy evaluation by ER using MD trajectory
- **(a)** + **(b)** FMO based MD/ER -> Free energy
  - a combination **FMO based MD** with **MD/ER**
  - Protein solvation as a whole is possible, but ligand binding scheme is now in development.

Focus on (a)FMO using GPU in the present talk

X-Ability
クロスアビリティ

Plan for implementation



GPGPU add-on is developed by us.

1. Use best tuned algorithm as far as GPU's resource permits.
2. Shared memory is used as L1 cache.
3. Use CPU threads (pthread) to harness GPU calculations for complicated thread control (synchronization, queuein..)
4. Use OpenMP for CPU thread parallelization.
5. Single/double mixed-precision calculation.

X-Ability
クロスアビリティ

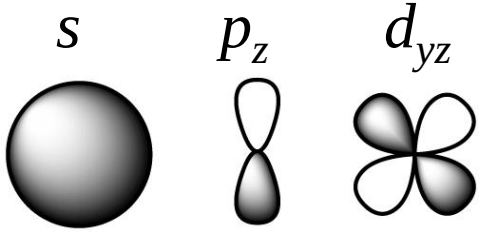
How to solve Schrödinger equation?

$$H(1\cdots N)\Psi(1\cdots N) = E\Psi(1\cdots N)$$

Expand wave function with

"contracted Gaussian basis set"

(Linear Combination of Atomic Orbital approximation)



$$\psi_i(r) = \sum_k C_k^{(i)} \chi_k(r)$$

$$s: \ \chi(r) = \sum_{k=1}^{K} d_k e^{-\alpha_k (r-A)^2} \qquad p_x: \ \chi(r) = \sum_{k=1}^{K} d_k (x - A_x) e^{-\alpha_k (r-A)^2}$$

$$F(C)C = SC\varepsilon$$

Matrix form of eigenvalue problem
(Self consistent)

X-Ability
クロスアビリティ

Hartree-Fock SCF (HF-SCF) Procedure

$$F(C)C = SC\varepsilon$$: SCF

$$(ab\,|\,cd) = \int \frac{\chi_a(r)\chi_b(r)\chi_c(r')\chi_d(r')}{|r-r'|}\,dr'\,dr$$: ERI

$$F_{\mu\nu} = H_{\mu\nu}^{core} + \sum_{a}^{2/N}\sum_{\lambda\sigma} C_{\lambda a}C_{\sigma a}^{*}\left[2(\mu\nu|\sigma\lambda) - (\mu\lambda|\sigma\nu)\right]$$

$$= H_{\mu\nu}^{core} + \sum_{\lambda\sigma} D_{\lambda\sigma}\left[(\mu\nu|\sigma\lambda) - \frac{1}{2}(\mu\lambda|\sigma\nu)\right]$$

$$= H_{\mu\nu}^{core} + G_{\mu\nu}$$

Density Matrix (D) : is updated at every SCF cycle to solve nonlinear equation after initial guess.

ERIs (ab|cd) : may be calculated only once and store them on memory in principle, but it needs large O(N^4) memory. They are recalculated at every cycle (direct SCF) to reduce expensive disk I/Os. **This step is bottleneck.**

Density Matrix × ERI = J-matrix : Coulomb potential matrix

Density Matrix × ERI = K-matrix : HF exchange matrix. **GPU implementation is not easy because it needs a lot of registers.**

X-Ability
クロスアビリティ

Procedure of J-matrix (Coulomb)

Construct shell-pair data $|ab]$, $|p]$, D_q

expand them in terms of primitive HGs

Sort $|p]$ and $|q]$ for faster exit of loops.
$|p]$: $[p|p]^{1/2}$
$|q]$: $[q|q]^{1/2}$ D_q
descending order

p, q, D_q

$$J_p = \sum_q [p \mid q] D_q$$

1. Each thread between each pair of $[p|q]$.

2. No Integral symmetry to avoid expensive memory access.

Accumulated J_p

converted to J_{ab}

Less than 10% of total cost

X-Ability
クロスアビリティ

What's FMO(Fragment MO)

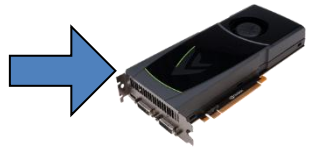
Ab initio for large Insulators (protein etc.)

Divide a molecule into N fragments

Generate initial density matrices for all fragments (monomers)

Prepare **environmental electrostatic potentials (ESP)** using previous density matrices

Bottleneck



Solve Fock equations for all monomers $F^I C^I = S^I C^I e^I$, for I=1 to N

NO

Are all monomer densities converged?

YES

Prepare environmental electrostatic potentials for all fragment pairs (dimers) using monomer densities

Solve Fock equations for all monomers $F^I C^I = S^I C^I e^I$, for I=2 to N, J=1 to I-1

Calculate total energy and properties of molecule

X-Ability
クロスアビリティ

Acceleration of Environmental Electrostatic Potential (ESP)

Fragment A Fragment B

$$J_{ab} = \sum_{cd} (ab|cd) D_{cd}$$

- Utilize ERI J-matrix

  cd: basis on neighboring fragments

- Decompose protein into many small fragments such as amino acid molecules.

  Conventional SCF (ERIs saved on disk) is effective for usual amino acids (# of basis < 180).

  ERI is not bottleneck in FMO.

  Environmental electrostatic potential dominates.

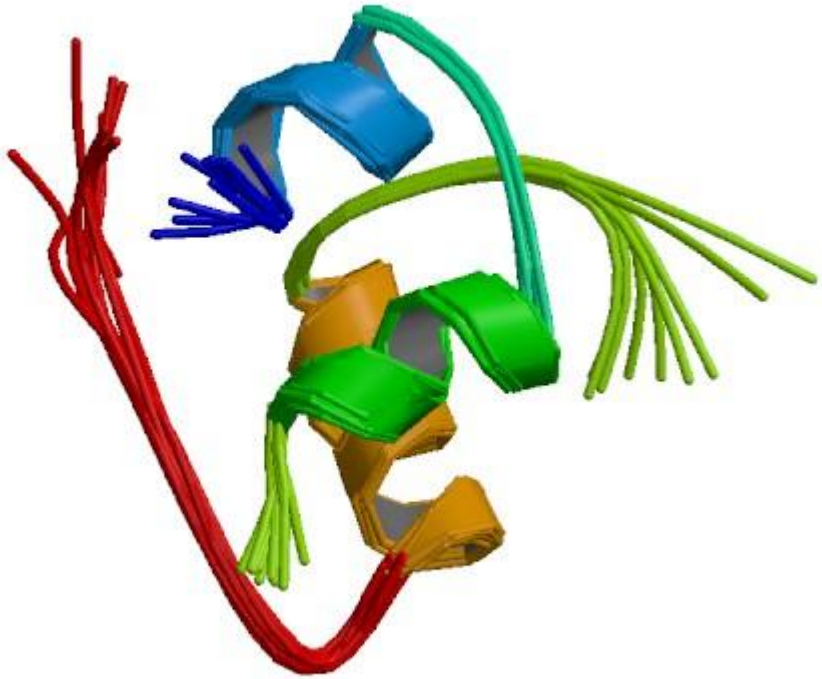
- Utilize J-matrix acceleration program by

X-Ability
クロスアビリティ

Test Model

Insulin (PDBID:2HIU)

Small Protein
44 amino acids



X-Ability
クロスアビリティ

System Configuration

(1) Intel Core i7-3930K @3.2GHz (6 core) ,
GTX580 x 2, 32GB, CUDA 4.0

(2) Intel Core i7-3930K @3.2GHz (6 core),
GTX580 x 4, 32GB, CUDA 4.0

(3) Intel Core i7-3930K @3.2GHz (6 core) ,
GTX680 x 2, 32GB, CUDA 4.0

+ Intel Composer XE 12.0 + MKL 10.3

X-Ability
クロスアビリティ

Total Energy of Insulin

	Time [sec]	Total Energy [a.u.]
Original GAMESS	3279.944	-21635.4488652520
Our GPGPU work	790.527	-21635.4488649211

(1) Intel Core i7-3930K @3.2GHz (6 core) , GTX580 x 2, 32GB, CUDA 4.0
(2) Intel Core i7-3930K @3.2GHz (6 core), GTX580 x 4, 32GB, CUDA 4.0
(1) + (2)

- Error of total energy of 44 fragments is small enough.
- Energy of each amino acid is highly consistent.

X-Ability
クロスアビリティ

FMO2-ESP and HF-SCF calculation time of Insulin in total computation

	GAMESS	This work	speedup
ESP part(GPU)	2571.490	170.897	x 15.0
HF-SCF part(host)	708.454	619.630	x 1.14
Total	3279.944	790.527	x 4.15

(1) Intel Core i7-3930K @3.2GHz (6 core) , GTX580 x 2, 32GB, CUDA 4.0
(2) Intel Core i7-3930K @3.2GHz (6 core), GTX580 x 4, 32GB, CUDA 4.0
(1) + (2)

ESP part is much better than total acceleration ratio.
HF-SCF part is on-memory calculation (faster than GPGPU).

X-Ability
クロスアビリティ

Comparison of architectures(FMO2 HF-SCF)

	Time [sec]	Total Energy [a.u.]
Original GAMESS	7026.264	-21635.4488652592
Our work [GTX580 x 2](1)	1670.931	-21635.4488649623
Our work [GTX680 x 2](3)	1708.522	-21635.4488649862

(1) Intel Core i7-3930K @3.2GHz (6 core) , GTX580 x 2, 32GB, CUDA 4.0
(3) Intel Core i7-3930K @3.2GHz (6 core) , GTX680 x 2, 32GB, CUDA 4.0

X-Ability
クロスアビリティ

FMO2 HF-SCF 6-31G* (d-orbital)

	ESP [sec]	Total time [sec]	Total energy [a.u.]
Original	19915.745	25360.740	-21643.9995562825
Our GPGPU work	1142.092	5839.737	-21643.9995952946
speedup	x 17.4	x 4.3	

(3) Intel Core i7-3930K @3.2GHz (6 core) , GTX680 x 2, 32GB, CUDA 4.0

X-Ability
クロスアビリティ

FMO2 HF-Gradient 6-31G

	ESP [sec]	Total time [sec]	Total energy [a.u.]
Original	13234.570	20222.516	-21635.4478224023
Our GPGPU work	1497.256	7917.169	-21635.4478217928
speedup	x 8.8	x 2.5	

(3) Intel Core i7-3930K @3.2GHz (6 core) , GTX680 x 2, 32GB, CUDA 4.0

Gradient is important for structure optimization.

X-Ability
クロスアビリティ

FMO3 HF-SCF 6-31G

	ESP [sec]	Total time [sec]	Total energy [a.u.]
Original	53898.693	72271.047	-21635.6075667898
Our GPGPU work	2589.916	19302.927	-21635.6075665173
speedup	x 20.8	x 3.7	

(3) Intel Core i7-3930K @3.2GHz (6 core) , GTX680 x 2, 32GB, CUDA 4.0

FMO3 is fmo calculation with many-body effect.

X-Ability
クロスアビリティ

Thank you for your attention.