REIN-K講習会

宮下　尚之
理化学研究所　神戸研究所
生命システム研究センター
分子機能シミュレーション研究チーム

創薬支援アプリ(ISLiMソフトウェア)講習会　２０１４年2月２６日
公益財団法人都市活力研究所 (NPO法人バイオグリッドセンター関西)

京コンピュータ

- 性能　10 PFlops　(LINPAC)
  June 2011 No.1(8.162PF),
  Nov. 2011 No.1(10.51PF),
  June 2012 No.2,
  Nov. 2012 No.3,
  June 2013 No.4

- 試験利用 from Apr. 2011 to Sep. 2012

- 88,128 node, 705,024 core (超並列スーパーコンピュータ），
  8 core/node (2GHz)

- 3D トーラスネットワーク

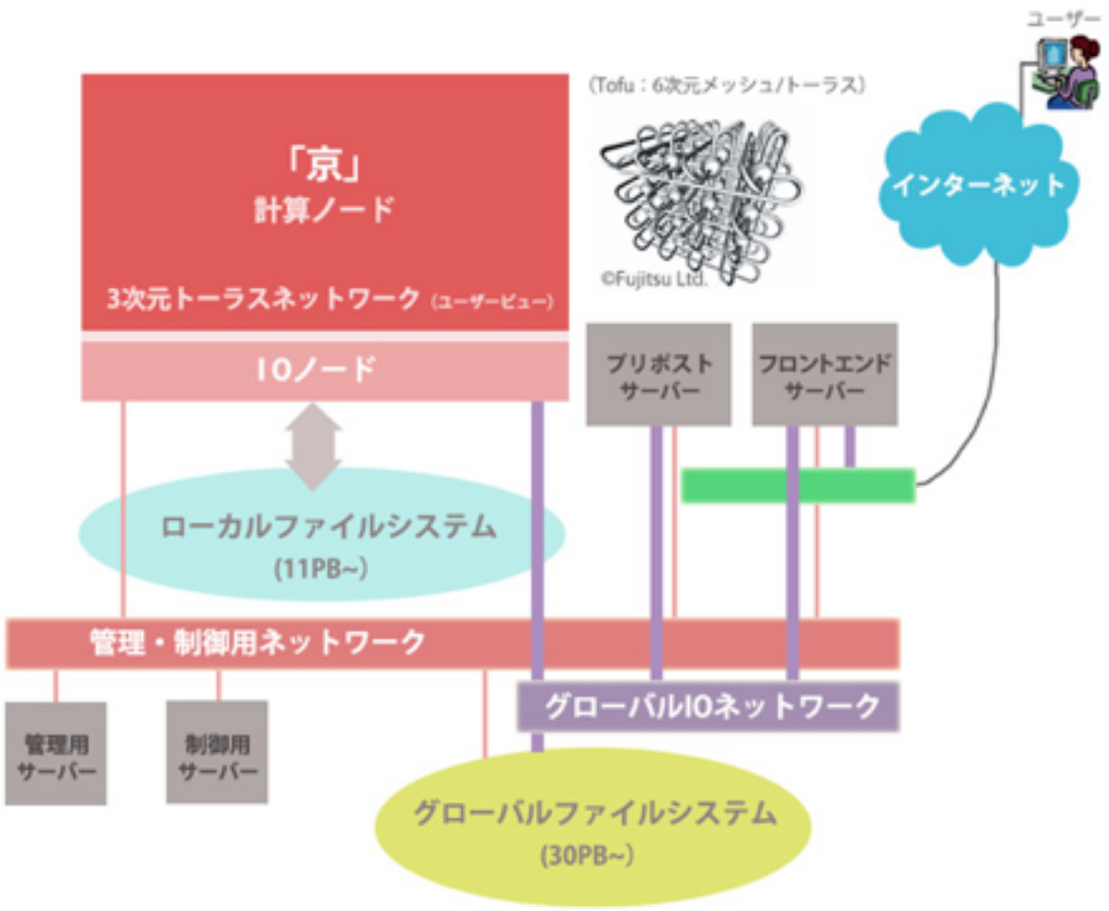
- SPARC, Fujitsu compiler (C, C++, Fortran), MPI-2, OpenMP

京
K computer

京コンピュータとSCLS FX10

	京コンピュータ	SCLS FX10
#Core/node	8 core	16 core
#Node	88,128 node	48 node
Total core	705,024 core	768 core
File system	Staging/Shared	Shared (home)
CPU	2.0GHz(sparc64VIIIfx)	1.6GHz(sparc64IXfx)
性能	10PF	10.1TF

京コンピュータのディスク



ステージングによるデータ転送

Frontend shared HDD

↓

計算ノード shared HDD

↓

X 通信不可

↑

計算ノード local HDD

動的プロセス生成では、shared HDDのみ利用可(REINで利用)

SCLS FX10ではステージングは無い

目次

多次元レプリカ交換インターフェースプログラム
Replica-Exchange INterface program for K
REIN-K

REIN プラットフォーム（計算科学）

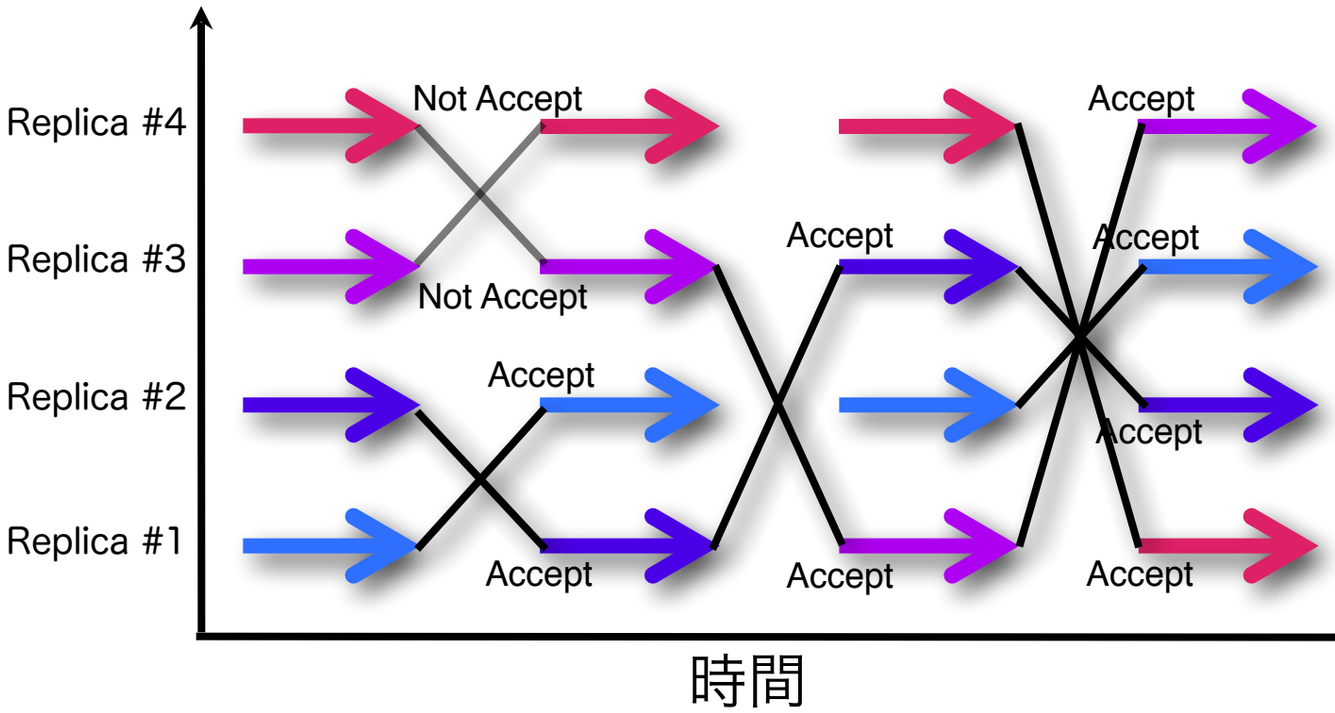
モデル	REINはbinary形式のプログラムを京コンピュータもしくはPCクラスタ上で複数起動させる
機能	任意のパラメータを変えて並列に起動し、ある周期毎に任意のパラメータの交換ができる
対応モジュール	京コンピュータで動くnamd（とmarble）に対応 PCクラスタで動くnamdに対応 インターフェースモジュールの開発により他にも対応
REMDの対応機能	現時点では、パラメータの交換対象は、温度とアンブレラポテンシャル
今後	将来的には対応するMDの種類と、レプリカ交換計算の種類を増やします。

利用目的・対応状況（分子シミュレーション）

機能	REINは既存の分子動力学(MD)プログラムを利用して、多次元レプリカ交換(MREM)シミュレーションを行う為のプログラム
対応機種	REINは京コンピュータ, FX10とPCクラスタに対応
対応スケジューラ	富士通のスケジューラ、Grid Engine
想定する研究	計算対象は、生体高分子の構造予測, 分子間の結合自由エネルギー計算、構造変化揺らぎの予測です。
利用するMDプログラム	現時点ではMDはNAMD2を利用します (MARBLEもKでは対応はしていますが、現時点ではNAMD2での利用を推奨します。)
レプリカ交換の対応機能	現状、温度レプリカ交換、距離、角度、二面角のレプリカ交換アンブレラサンプリングに対応
今後	将来的には対応するMDの種類と、レプリカ交換計算の種類、対応機種・スケジューラを増やします。

レプリカ交換法の基礎

Replica-exchange molecular dynamics (REMD)



・同じシステムを複数用意（レプリカ）
・異なる温度（パラメータ）を与える
・MDを実行
・あるステップ毎にメトロポリスクライテリアに従って隣合う温度を持つレプリカ間で交換評価

- $H(p_i,q_i) = K(p_i) + E_0(q_i)+U(q_i)$
- Transition probability:

Metropolis criteria : $\min[1, \exp(-\Delta)]$

- Exchange

$$\Delta_x = \beta_m(U_m(q_j)-U_m(q_i))-\beta_n(U_n(q_j)-U_n(q_i))$$

[REM] K. Hukushima, and K. Nemoto, J. Phys. Soc. Jpn. 65, 1604-1608 (1996)

[1] Y. Sugita and Y. Okamoto, Chem. Phys. Lett. 314, 141-151, (1999)

[2] Y. Sugita, A. Kitao and Y. Okamoto, J. Chem. Phys. 113, 6042-6051 (2000)

MCMC

普通のモンテカルロ法：
静的モンテカルロ法

$$X_1 \rightarrow X_2 \rightarrow \ldots\ldots \rightarrow X_i \rightarrow X_{i+1} \rightarrow$$

・マルコフ連鎖モンテカルロ（動的モンテカルロ法）
　・手順：乱数をふるが、一つ手前の情報を用いて、次の状態を得る方法

MCMCの条件

- どんな初期分布P_0から初期状態x_0を選んでも、$t \to \infty$で$Pt(x)$は定常分布$P(x)$に収束する。

詳細釣合

- 先の条件を満たす条件
- 遷移確率R(x|y): 状態yからxを生成

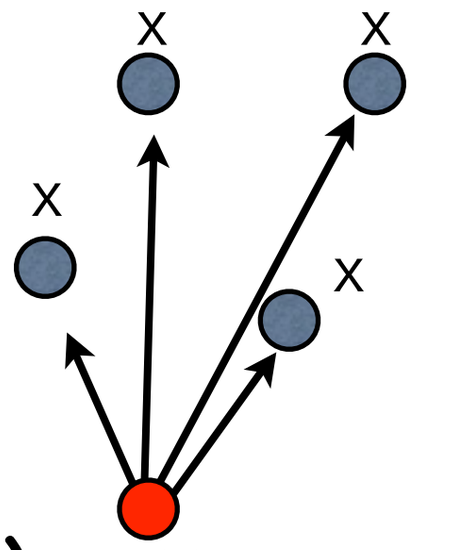
$$R(x|y)f(y) = R(y|x)f(x)$$

R(x|y)f(y)：事前確率

メトロポリス法

$x_1 \rightarrow x_2 \rightarrow \cdots\cdots \rightarrow x_i \rightarrow x_{i+1} \rightarrow$

同じ確率ならT(x|y)=1/4



- 手順
- 候補を選ぶ（y→x）　　Trial: T(x|y)

$$\int dx T(x \mid y) = 1$$

- 受理するかどうかを決める
  受理の確率(Acceptance Prob.): A(x|y)

遷移確率：R(x|y)=A(x|y)T(x|y)

A(x|y)T(x|y)f(y)=A(y|x)T(x|y)f(x)

$$A(x|y)=T(x|y)f(y)=A(y|x)T(x|y)f(x)$$

$$A(x \mid y) = \frac{T(y \mid x) f(x)}{T(x \mid y) f(y)} A(y \mid x)$$

$$A(x|y)=r(x|y)A(y|x)$$

A(x|y)=1　=>　A(y|x)<1、r(x|y)>1

A(y|x)=1　=>　A(x|y)<1，r(x|y)<1

この場合A(x|y)=r(x|y)

Acceptance Probability:A(x|y)=min[1,r(x|y)]

r(x|y)>1の時は、必ず受理。r(x|y)<1なら、乱数をふって、r(x|y)>randなら受理、r(x|y)<randなら不採用。

レプリカ交換法の受理確率

今、M個のパラメータで、M個のレプリカが
あるとする。全体の分配関数は、

$$Z = \Pi_{m=1}^{M} Z(\beta_m)$$

確率分布関数は

$$f(X;\beta) = \Pi_{m=1}^{M} \frac{\exp(-\beta_m H(X_m))}{Z(\beta_m)} \qquad \beta = 1/k_B T$$

詳細釣合の式を思い出す。

R(x|y)f(y) = R(y|x)f(x)

$$r(x \mid y) = \frac{T(y \mid x) f(x)}{T(x \mid y) f(y)}$$

隣り合うパラメータ間の交換は、T(x|y)=T(x|y)

$$A(\mathbf{x}|\mathbf{y}) = \min\left[1, \frac{f(\mathbf{x})}{f(\mathbf{y})}\right].$$

今、H(p,q)=K(p)+U(q), ただしpとK(p)はMDの時
ここではMCの時を仮定。H(x)=U(x)とする。

$$\frac{f(x)}{f(y)} = \frac{\exp(-\beta_n H(y_n))\exp(-\beta_m H(x_m))}{\exp(-\beta_m H(y_m))\exp(-\beta_n H(x_n))}$$
$$= \exp(-\beta_n (H(y_n) - H(x_n)) - \beta_m (H(x_m) - H(y_m)))$$
$$= \exp(-\beta_n (U(y) - U(x)) + \beta_m (U(y) - U(x)))$$
$$= \exp[-(\beta_n - \beta_m)(U(y) - U(x))]$$
$$= \exp[-\Delta],$$
$$\Delta = (\beta_n - \beta_m)(U(y) - U(x))$$

A(y|x)=min[1,exp(-Δ)]

尚、MDの時は温度スケーリングを考慮すること。

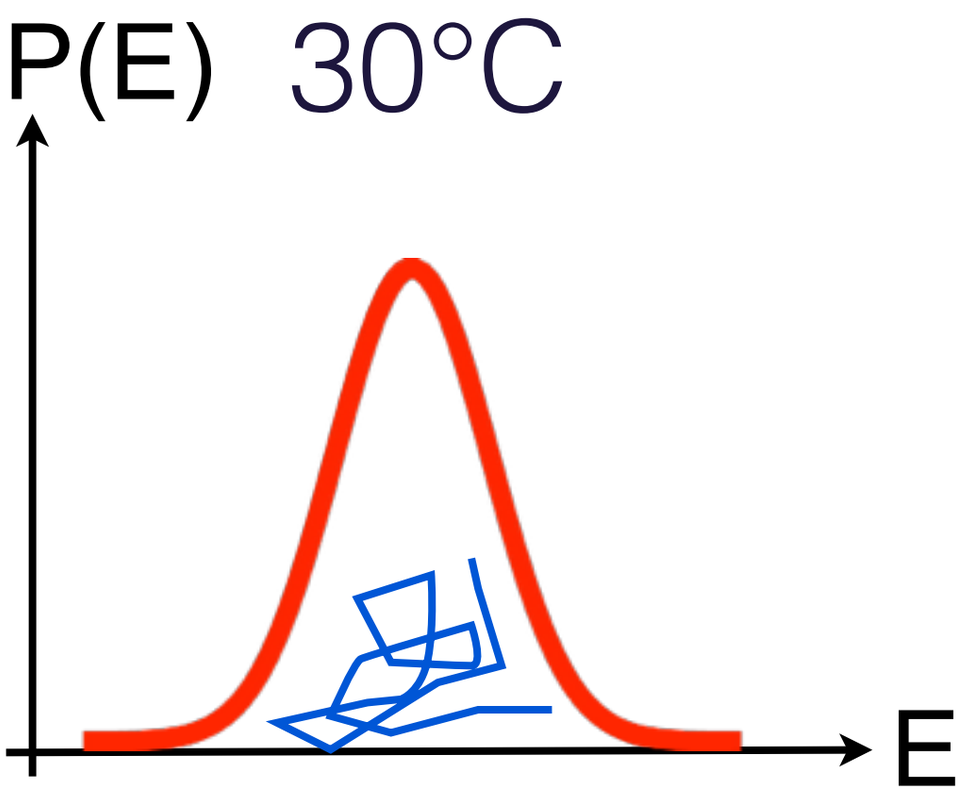
$$\langle K(p) \rangle_T = \left\langle \sum_{k=1}^{N} \frac{p_k^2}{2m_k} \right\rangle_T = \frac{3}{2} N k_B T .$$

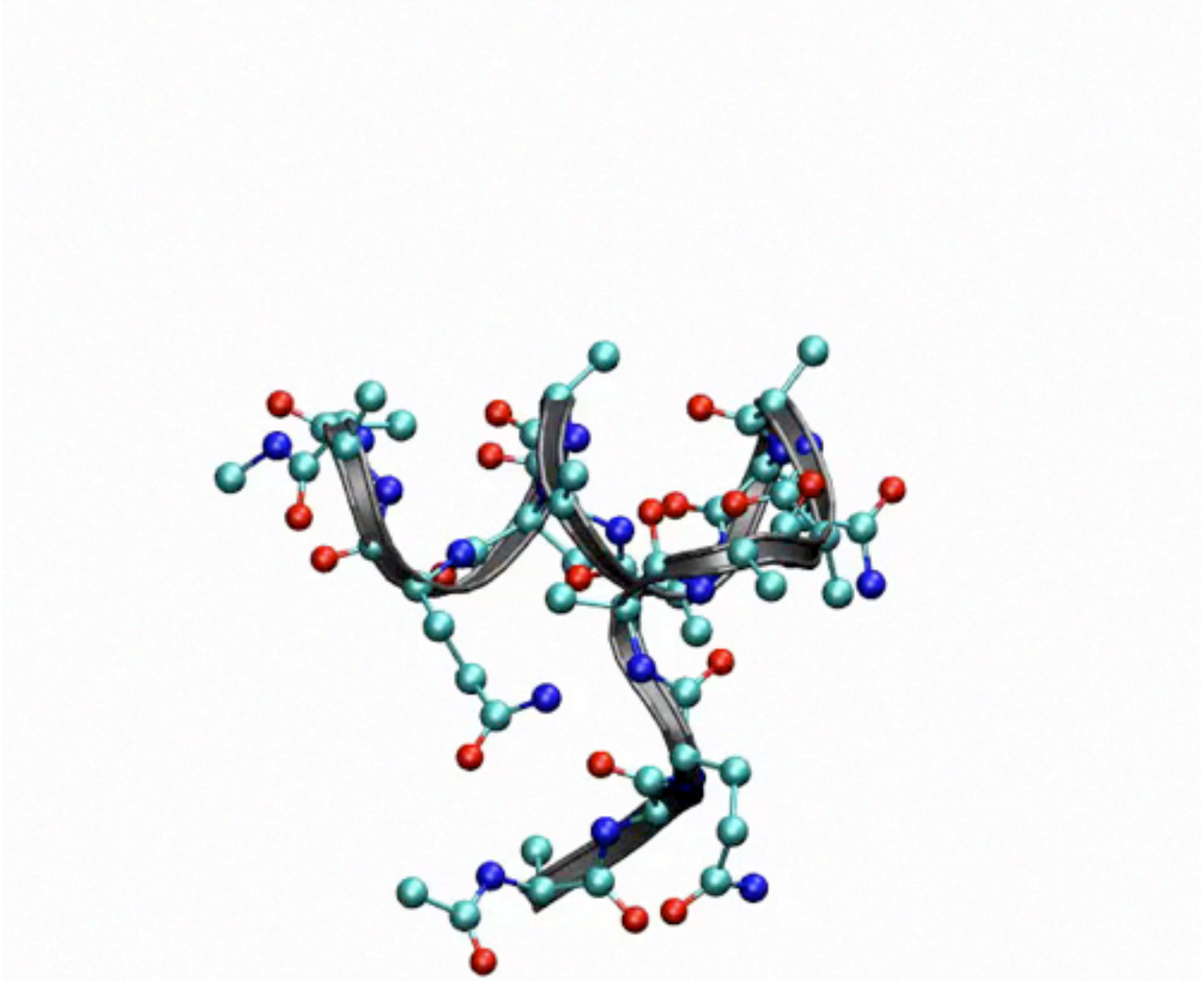
$$\begin{cases} p^{[i]'} \equiv \sqrt{\frac{T_n}{T_m}}\, p^{[i]}, \\ p^{[j]'} \equiv \sqrt{\frac{T_m}{T_n}}\, p^{[j]}, \end{cases}$$

例：タンパク質の構造予測

通常のMDでは構造探索あまりできない

通常の分子動力学法（MD）

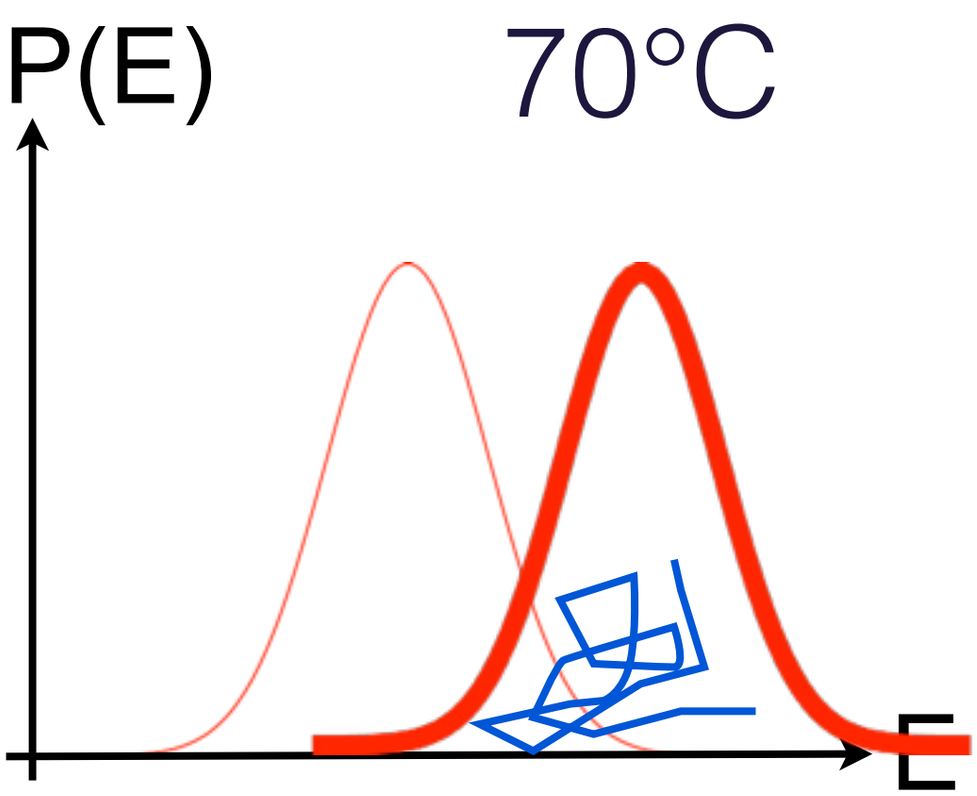


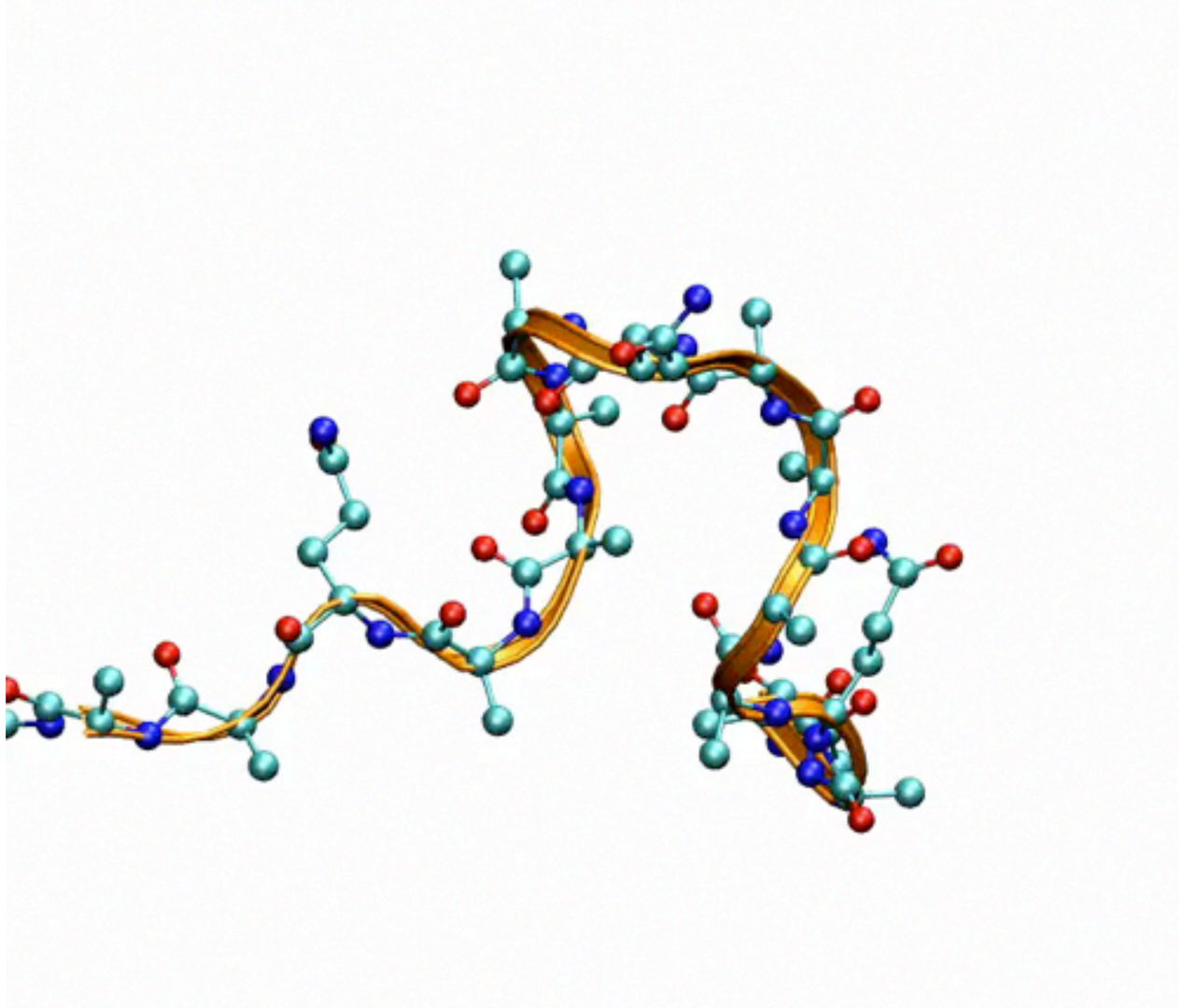


エネルギー空間、局所最小値に捕まる

正しい構造が見つからない

通常のMDでも高温だといろいろな構造をサンプルできる。しかし

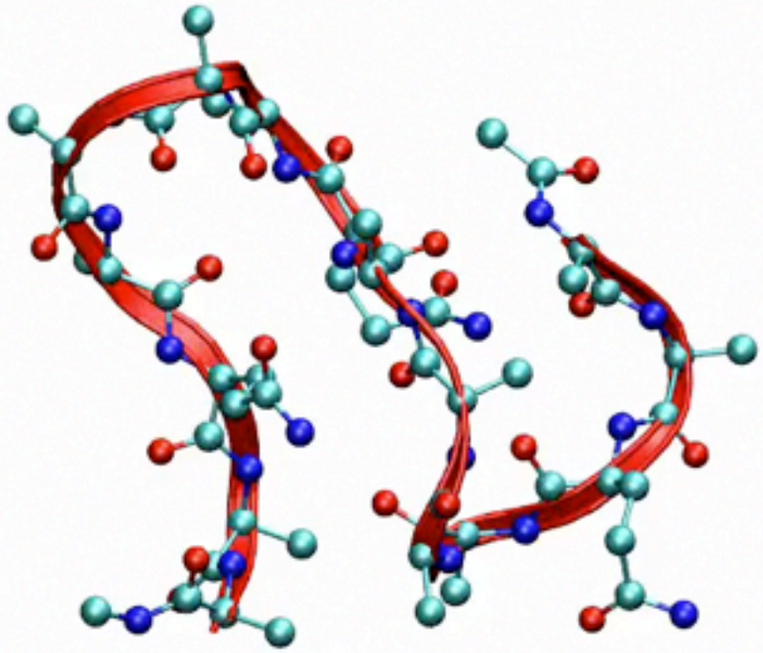


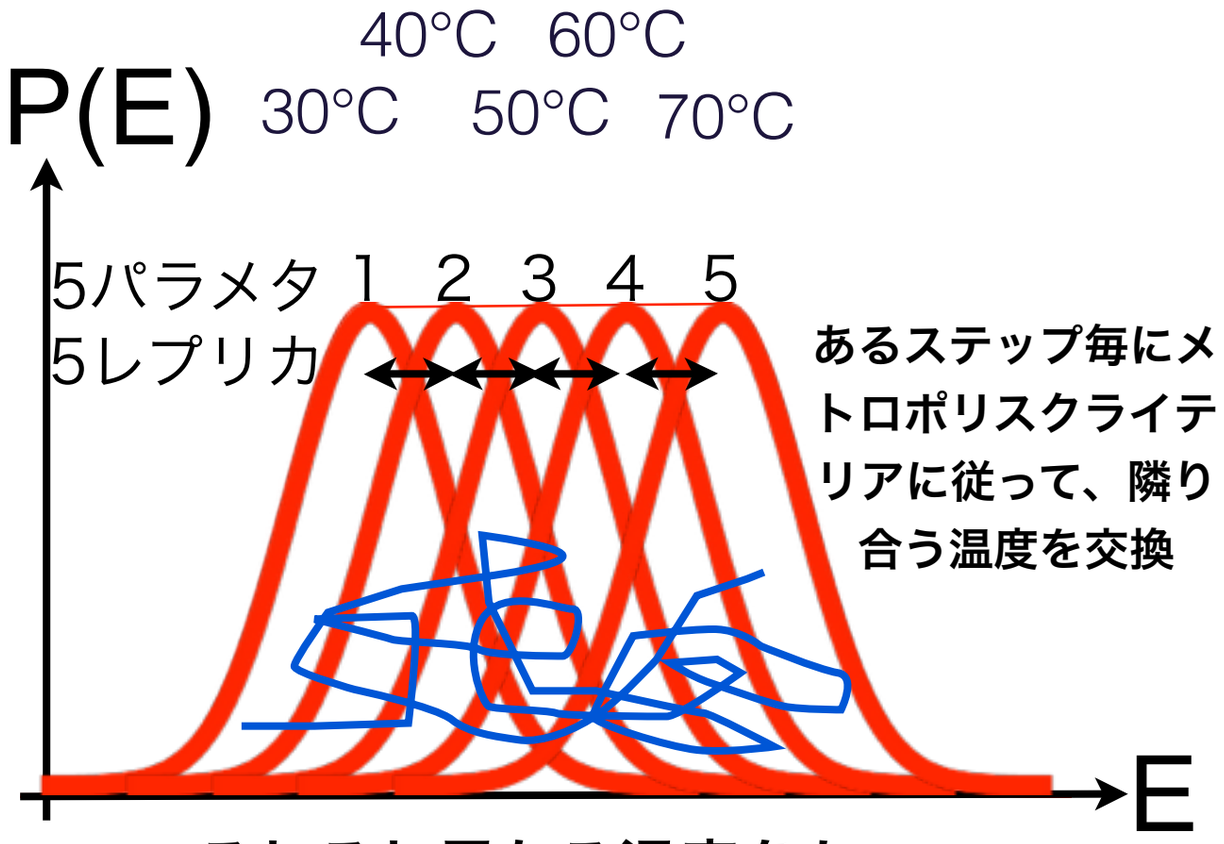


エネルギー空間、局所最小値に捕まらないが

知りたい温度の構造が得られない

レプリカ交換分子動力学法(REMD)
温度上のランダムウォーク





それぞれ異なる温度をもつ

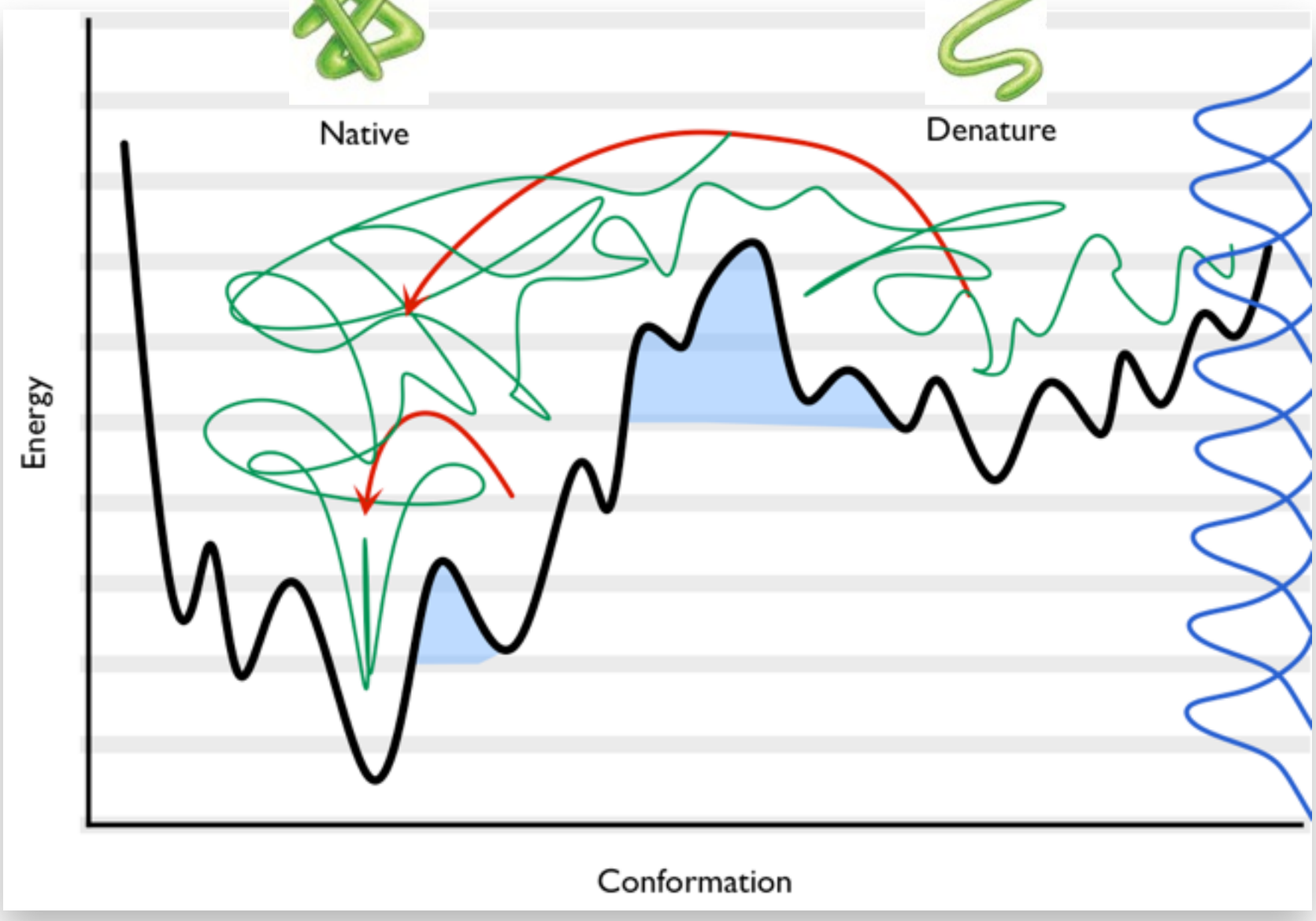
複数のレプリカを用意しMD計算

エネルギー空間、構造空間を幅広くサンプリングできる

統計処理を行い知りたい温度での構造決定

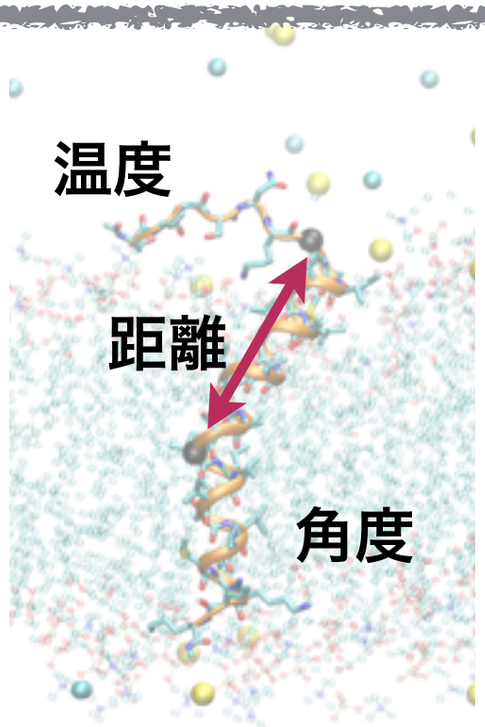
バリアを乗越える事ができる

温度の異なるMDを行い、ピコ秒毎に温度を交換

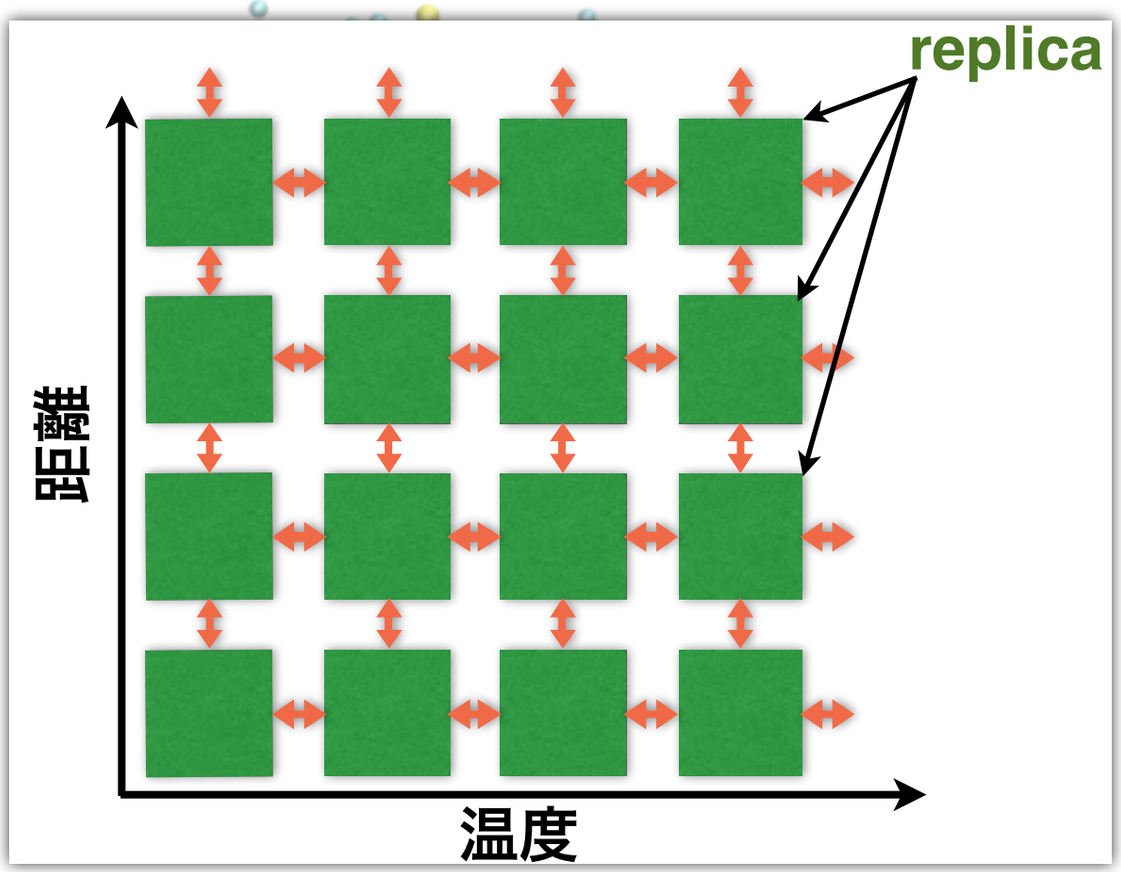


@100℃
@90℃
@80℃
@70℃
@60℃
@50℃
@40℃
@30℃

多次元レプリカ交換法(MREM)



- 温度だけでなく、距離や角度、二面角、その他のパラメータ交換にも拡張可能。
  たとえばReplica-exchange umbrella sampling (REUS)[2]
- それらを組み合わせて多次元レプリカ交換
  Multi-dimensional REMD (MREM) [2]
- ただし、レプリカ数が増える



[2] Y. Sugita, A. Kitao and Y. Okamoto, J. Chem. Phys. 113, 6042-6051 (2000)

京コンピュータの沢山のノードを利用した計算が実現できる。

レプリカ交換法を用いたアプリケーション例

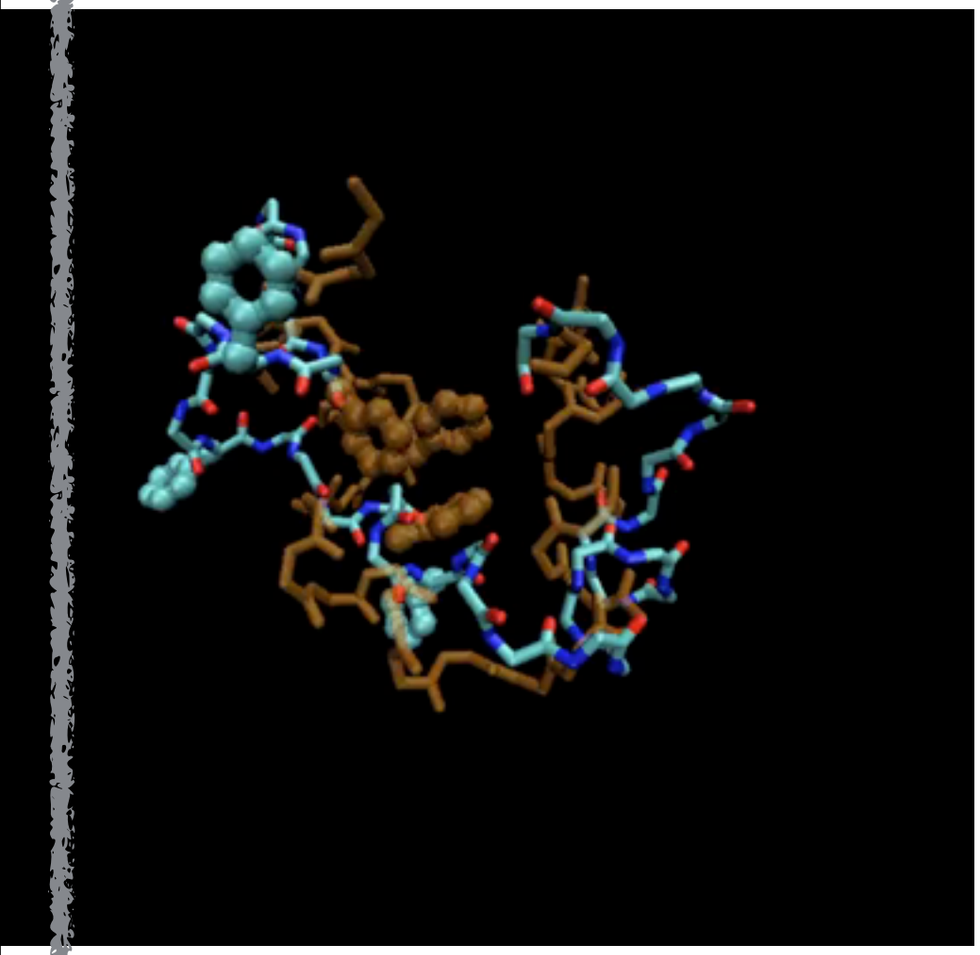
Villin HP36のフォールディング

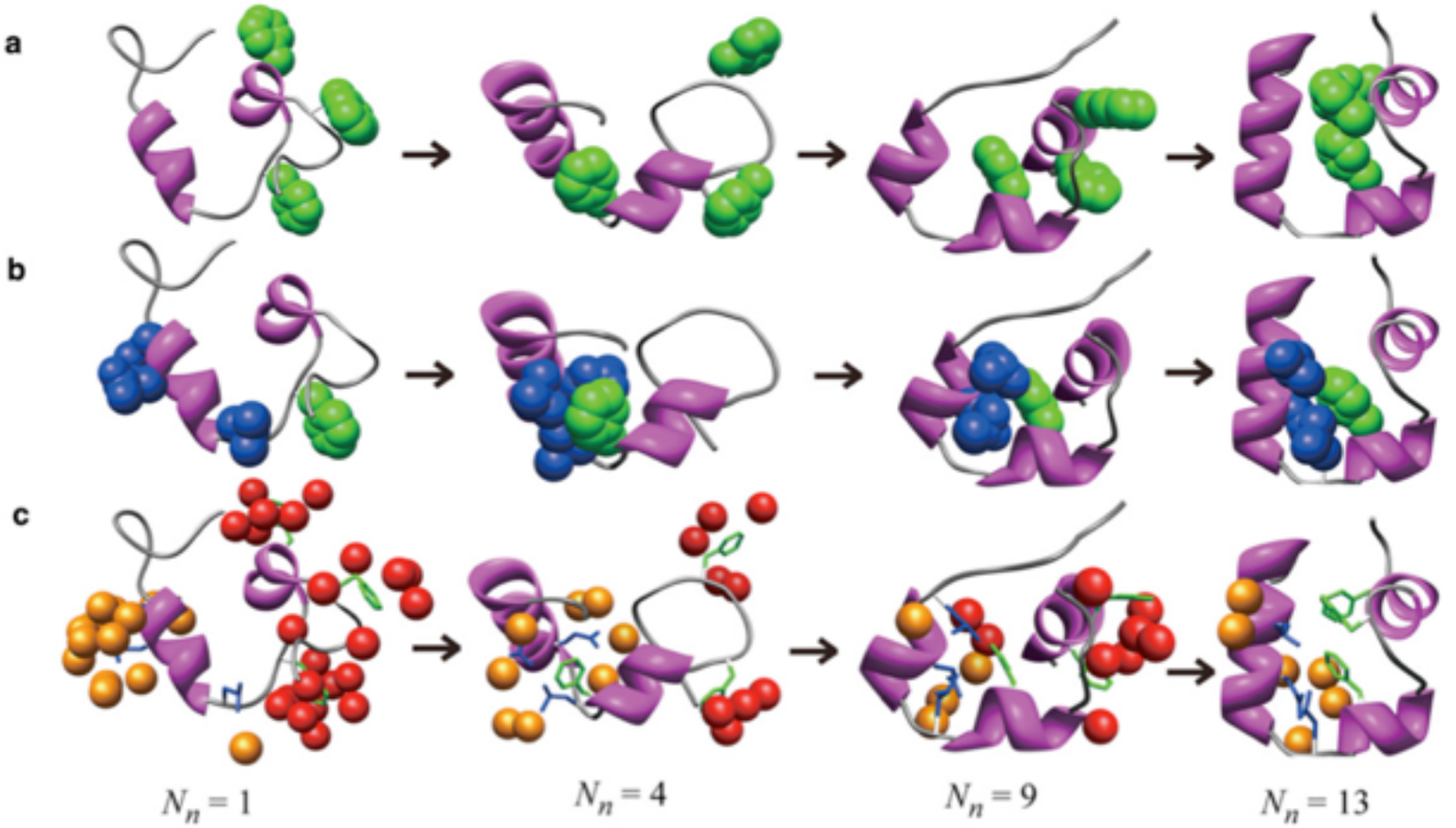
REMDとMulticanonicalを組み合せた方法

Yoda, T., Sugitate, Y. & Okamoto, Y., Hydrophobic core formation and dehydration in protein folding studied by generalized-ensemble simulations. *Biophysical Journal*, 99(5), pp. 1637–1644. (2010)

依田先生
（長浜バイオ大）

hydrophobic coreの形成の際に水が抜ける

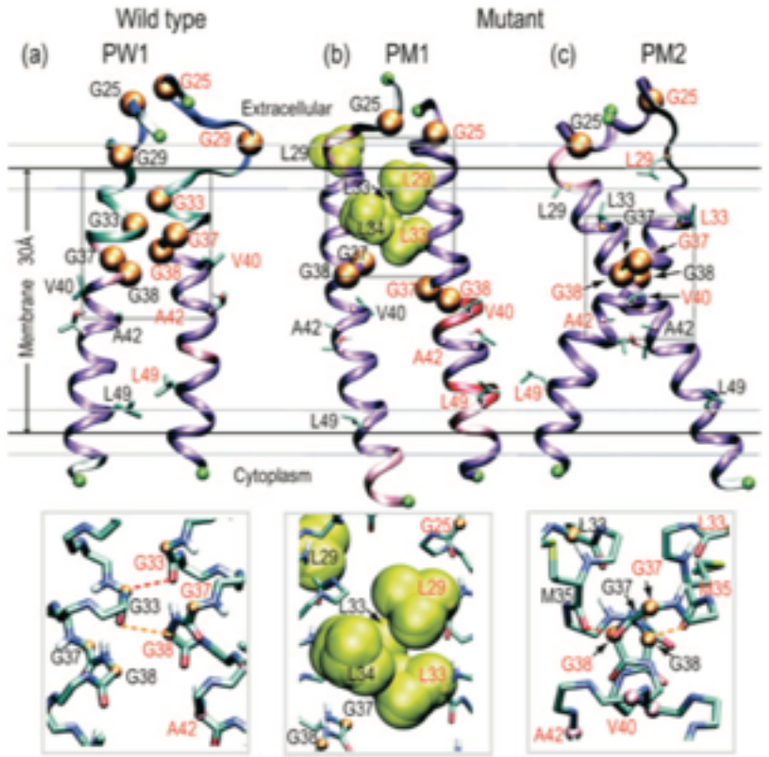


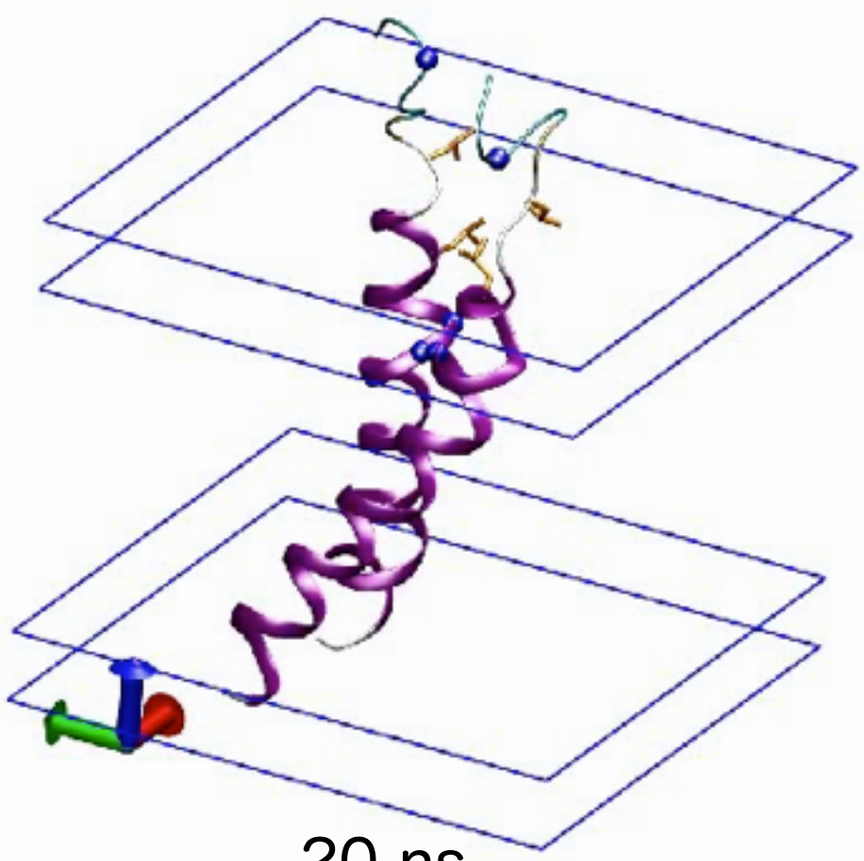


膜タンパク質の二量体構造予測

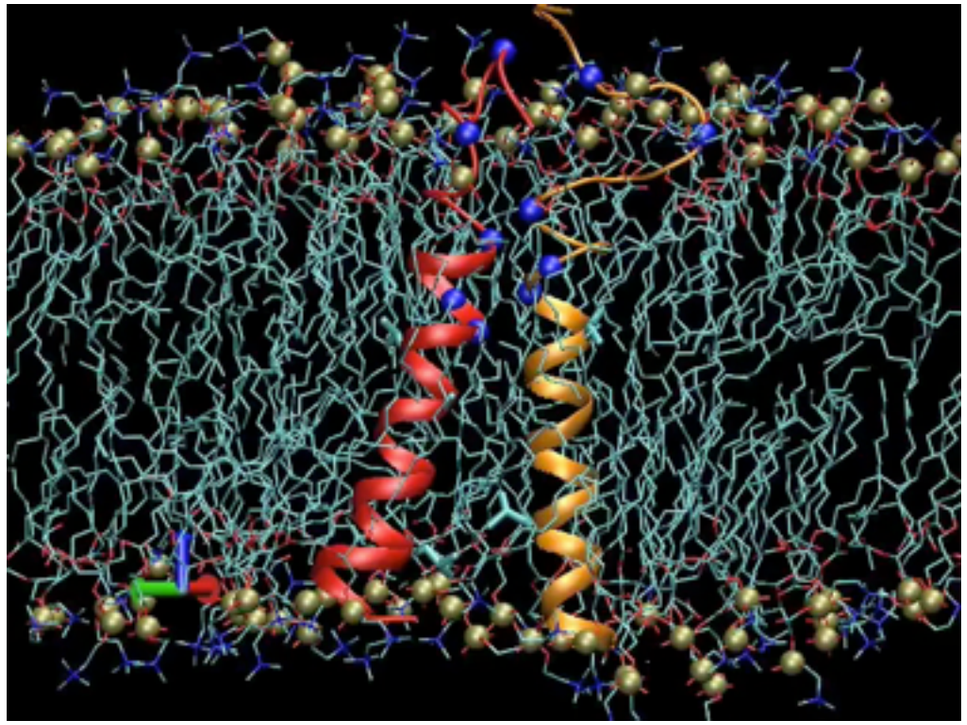
アミロイド前駆体タンパク質の二量体構造予測
インターフェースの変化がAβペプチド生成に影響

Naoyuki Miyashita, John E Straub, D Thirumalai, and Yuji Sugita, Transmembrane structures of amyloid precursor protein dimer predicted by replica-exchange molecular dynamics simulations. *Journal of the American Chemical Society*, 131(10), pp.3438–3439 (2009)





20 ns



30 ns

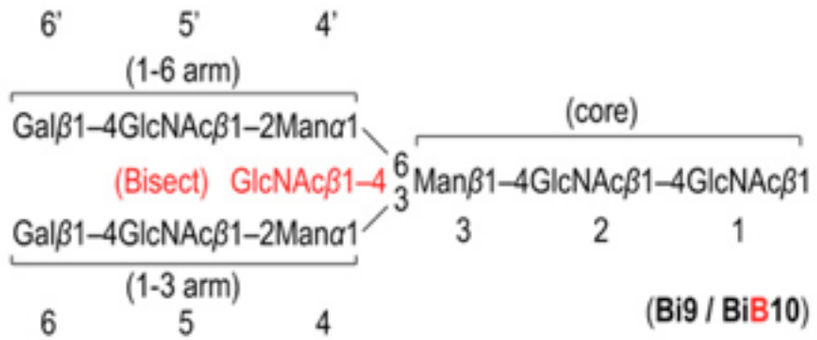
糖鎖分子の構造予測

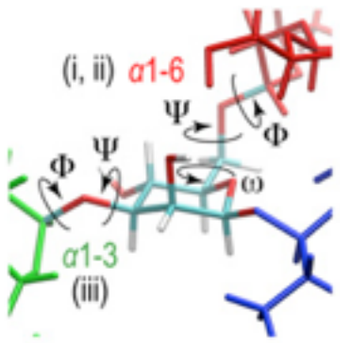
Suyong Re, Naoyuki Miyashita, Yoshiki Yamaguchi, Yuji Sugita, Structural Diversity and Changes in Conformational Equilibria of Biantennary Complex-Type N-Glycans in Water Revealed by Replica-Exchange Molecular Dynamics Simulation,

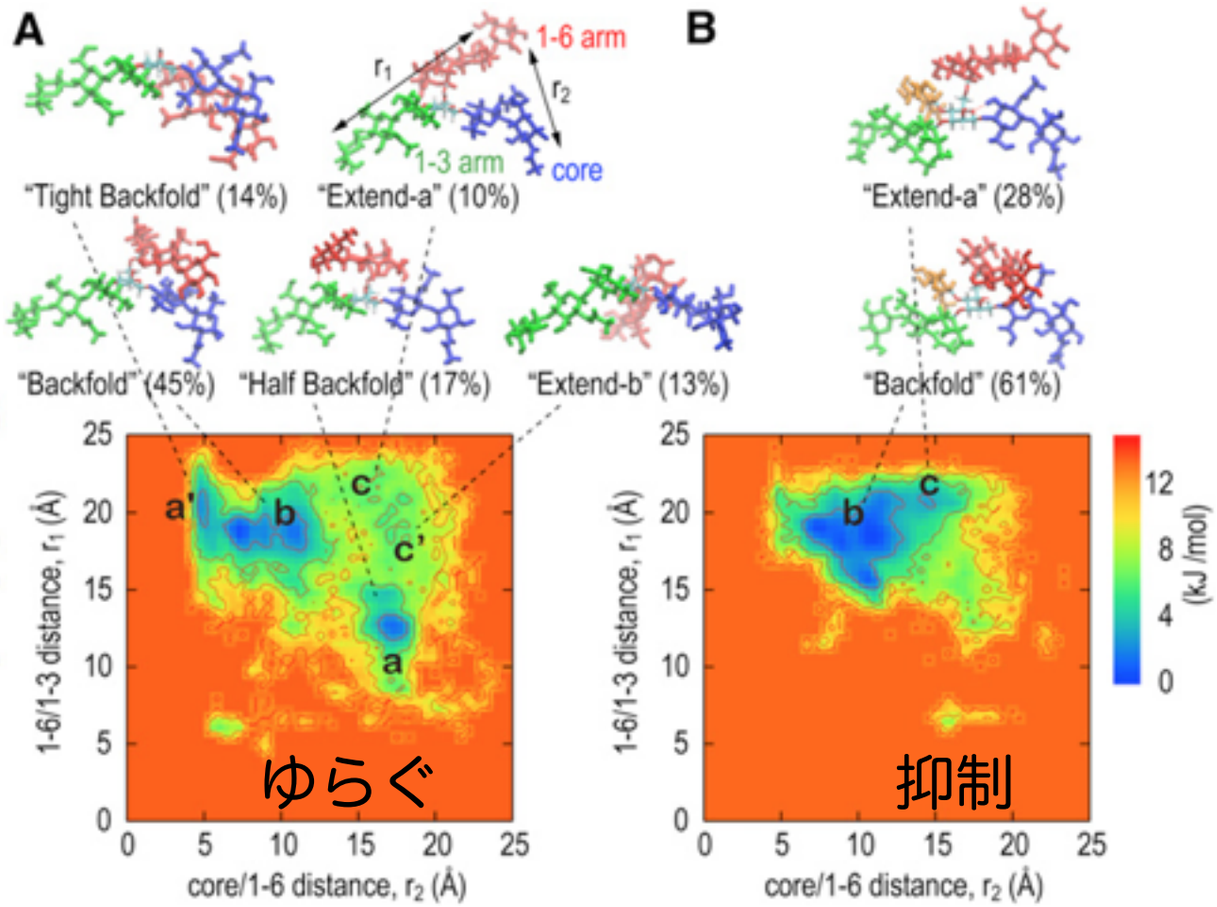
Biophys. J. **101**, Pages L44–L46 (2011)

李博士(理研)

REINでのREMD計算で糖鎖（N-glycan）の構造予測を行った。分子に付いた糖鎖の揺らぎは分子間の認識に重要である。実験の予想どおり bisecting GlcNAcは揺らぎにおおきな影響を及ぼしている事がわかった。







bisect無し　bisectあり

カポジウィルス由来タンパク質の免疫システムへの攻撃

実験との共同研究

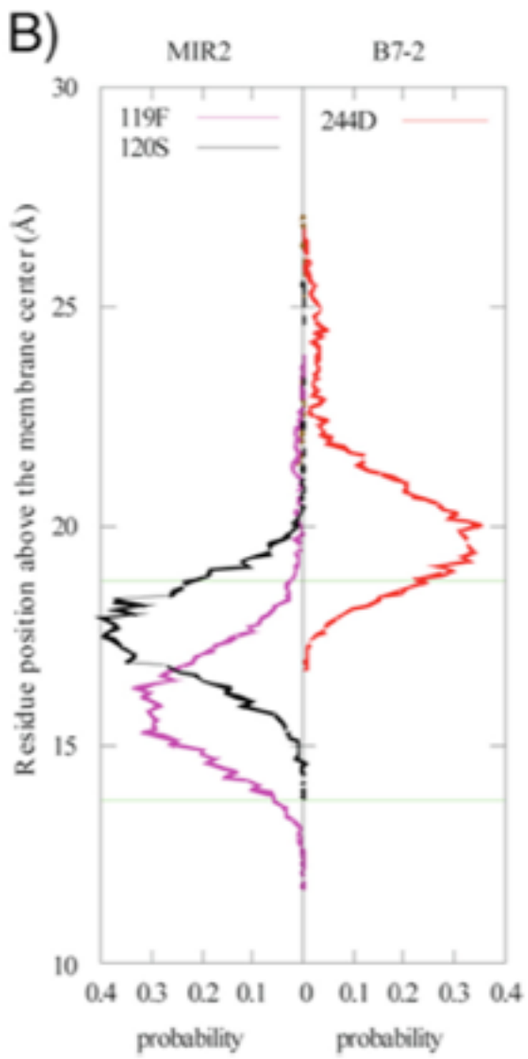
梶川博士(昭和薬大)

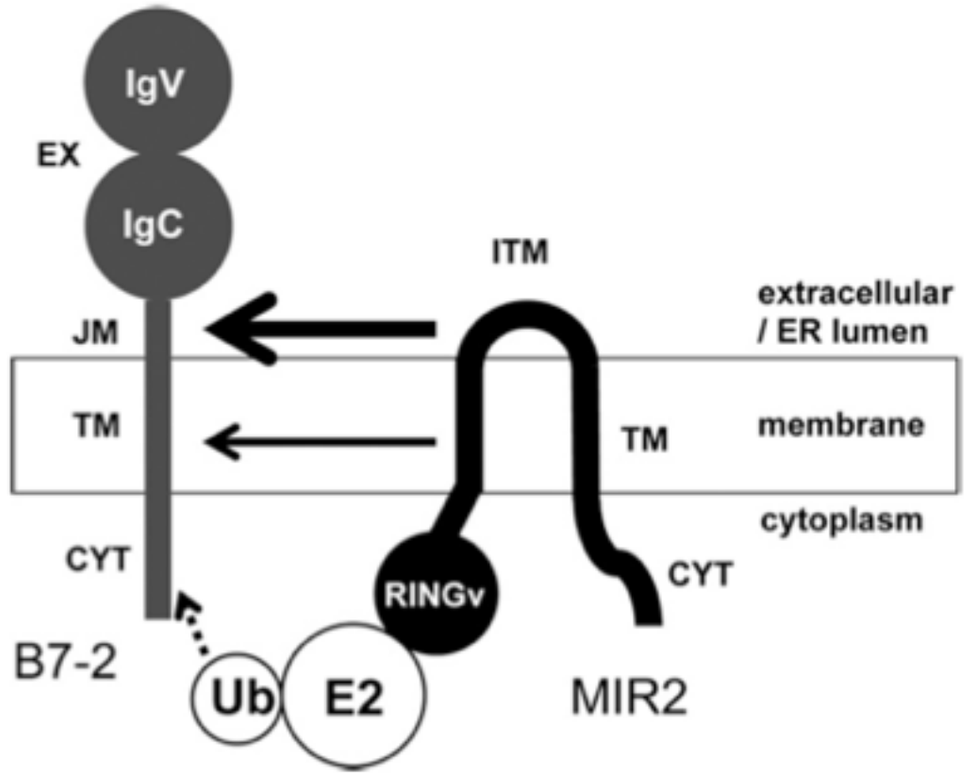
Pai-chi Lee博士（理研）

Mizuho Kajikawa, Pai-Chi Li, Eiji Goto, Naoyuki Miyashita, Masami Aoki-Kawasumi, Mari Mito-Yoshida, Mika Ikegaya, Yuji Sugitate, and Satoshi Ishido,
The intertransmembrane region of Kaposi's sarcoma-associated herpesvirus modulator of immune recognition 2 contributes to B7-2 downregulation. *Journal of Virology*, 86(9), pp.5288–5296 (2012)

A)

B)





タンパク質リガンド結合　最近の論文

リガンドータンパク間距離と、溶媒温度の2次元レプリカ交換分子動力学シミュレーション

Two-Dimensional Replica-Exchange Method for Predicting Protein–Ligand Binding Structures

Hironori Kokubo, Toshimasa Tanaka, and Yuko Okamoto,

Journal of Computational Chemistry 2013, 34, 2601-2614 (2013)

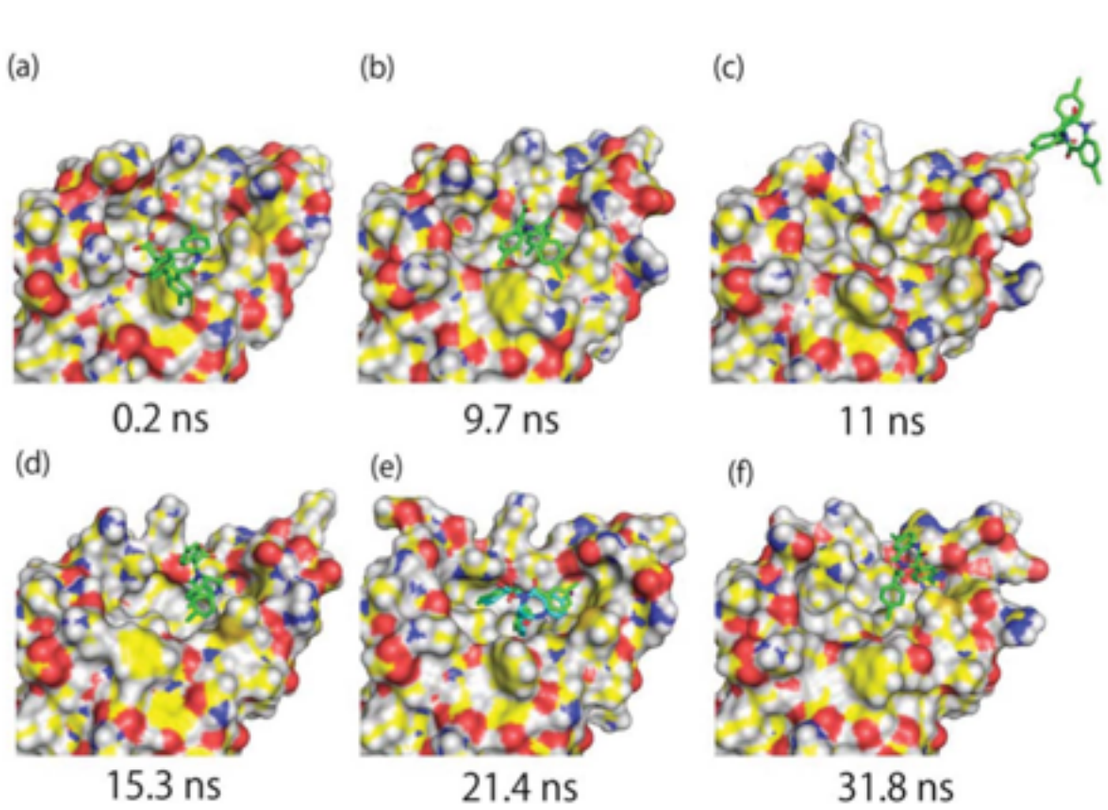
小久保博士（Takeda Pharmo.）

$$w(X \to X') = \begin{cases} 1, & \text{for } \Delta \le 0, \\ \exp(-\Delta), & \text{for } \Delta > 0, \end{cases} \quad (9)$$

where

$$\Delta = \beta_0 \left(E_{m_2}\left(q^{[j]}\right) - E_{m_2}\left(q^{[i]}\right) - E_{m_2+1}\left(q^{[j]}\right) + E_{m_2+1}\left(q^{[i]}\right) \right) \quad (10)$$

$$= \left(\beta_{m_2} - \beta_{m_2+1}\right) \left(U_{II}\left(q^{[j]}\right) - U_{II}\left(q^{[i]}\right) + \frac{\sqrt{\beta_0}}{\sqrt{\beta_{m_2}} + \sqrt{\beta_{m_2+1}}} \left(U_{Is}\left(q^{[j]}\right) - U_{Is}\left(q^{[i]}\right) \right) \right) \quad (11)$$



1T4EとBenzodiazephineの結合

REIN-Kとは

よく使われる MD パッケージ

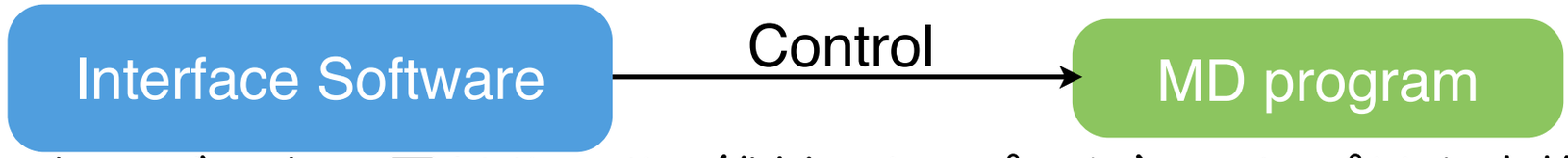
MDパッケージ	Hybrid	Speed	REMD	MREM	on K
CHARMM [1]	×	Slow	○	○	×
AMBER [2]	×	Medium/Fast	○	×	×
NAMD [3]	○	Fast	○	○	△*
GROMACS [4]	○	Fast	○	○	△
DESMOND [5]	○	Fast	○	×	×
TINKER [6]	×	Slow	×	×	×
MARBLE [7]	○	Fast	×	×	○ (Native)

[1] B. R. BROOKS et al., J Comput. Chem 30: 1545–1614, 2009
[2] D. A. Pearlman et al., Comput. Phys. Comm. 91, 1-41, 1995
[3] J. C. Phillips et al., J. Comput. Chem. 26:1781-1802, 2005
[4] H.J.C. Berendsen et al., Comput. Phys. Comm. 91:43-56, 1995
[5] K. J. Bowers et al., In SC '06: Proceedings of the 2006 ACM/IEEE conference on Supercomputing, New York, NY,USA, 2006. ACM Press.
[6] J. W. Ponder and F. M. Richards, J. Comput. Chem., 8, 1016-1024, 1987.
[7] M. Ikeguchi, J Comput Chem 25: 529–541, 2004

- **REMD/MREMの使い勝手が悪い**
- **Kに未対応**

*** NAMD flat MPI版を京コンピュータにインストールした。hybrid(MPI/pthread)版はAICS佐藤チーム （神戸大）鎌田准教授と共同でインストールした。**

REIN以外のレプリカ交換法のインターフェースプログラム



異なるMDパッケージでも、同じルール（似たインプット）でレプリカ交換計算ができる。

パッケージ	MD	言語	MDの制御	MDの修正	MREM	K対応
aarex.pl in MMTSB toolsets [1]	CHARMM AMBER NAMD	Perl/ TCP socket	binary run (fork, exec function)	no	○	×
PLUMED [2]	GROMACS NAMD DLPOLY AMBER ACEMD LAMMPS QUANTUM- ESPRESSO CPMD	python/ (patch: depend on each MD)	Plugin style (It needs recompile with patch)	Necessary	○	×

[1] M. Feig et al., Journal of Molecular Graphics and Modelling 22 (2004) 377–395
[2] M. Bonomi, D. Branduardi, G. Bussi, C. Camilloni, D. Provasi, P. Raiteri, D. Donadio, F. Marinelli, F. Pietrucci, R.A. Broglia and M. Parrinello, PLUMED: a portable plugin for free energy calculations with molecular dynamics, Comp. Phys. Comm. 180, 1961 (2009), Available also on arXiv: 0902.0874

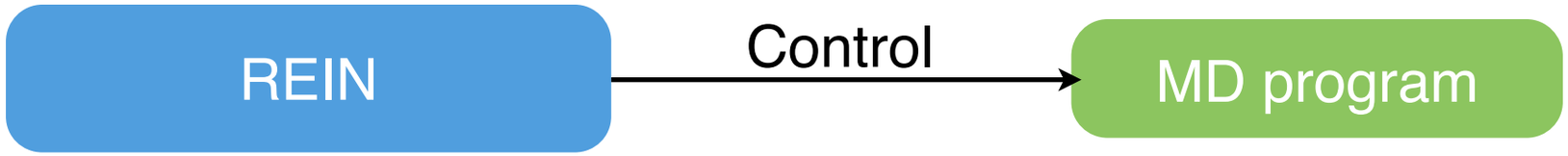
- **Advantage：　柔軟性が高い**
  - **→ それぞれのMD packageの機能が使える**
  - **→ REMD/MREMのインプットが簡単**
  - **→ 拡張が簡単**
- **Kでは動かない**

レプリカ交換インターフェースプログラム(REIN)

MD部分の修正不要	バイナリ形式の外部プログラム（既存のMDプログラムなど）を用いる（i/oのみ参照）
高速な並列化	PCクラスタでも京コンピュータでも動く（Laptopでも） 並列化はMPI/OpenMP
簡単実行	基本的なREMDの実行が簡単にできる（サポーティングソフトウエア）
詳細設定	交換の仕方などの細かい設定もできる。
モジュール化	モジュールとして他の外部MDプログラムの追加、機能追加ができる。
テンプレート	インプットやバッチファイルの設定を自動で行なうサポーティングソフトウェアでは、テンプレートを利用。

Replica-exchange Interface Program (REIN)

多次元レプリカ交換シミュレーションができる



パッケージ	MD	言語	MDの制御	MDのコード修正	MREM	K対応
REIN version 1.0.4	NAMD MARBLE	FORTRAN95, 03/MPI/ OpenMP	binary run [**mpi_comm_spawn** or **system function**]	no	○	○

MARBLE: Optimized on K.

NAMD: Currently flat MPI version are available on K.

REIN:

Machine	K computer (sparc), FX10 (sparc), PC cluster (x86_64), RICC(富士通のスパコン)	
Compiler	Fortran (Intel, fujitsu, (gfortran))	MPI-2 (Open MPI, Fujitsu MPI)/MPI-

多次元レプリカ交換（MREM）：

軸	REMD	REUS					
	温度	距離		角度		二面角	
		Atom	Group	Atom	Group	Atom	Group
NAMD	○	○	○	○	○	○	○
MARBLE	○	○	○	○	×	○	×

REIN-Kのしくみ

REINパッケージの構成

- rein
- supporting software
  - input_builder
  - batch_builder
  - remd_convert

REINは既存のバイナリの外部プログラム(MD)を子プログラムとして複数同時に起動する

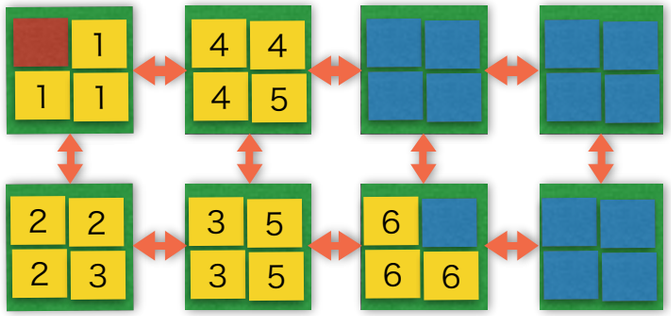
２つの仕組み	spawn	no spawn
利用技術	MPI_COMM_SPAWN	SYSTEM関数
対象コンピュータ	超並列コンピュータ K, FX10, PCクラスタ	PCやPCクラスタ、小規模並列コンピュータ、GPGPU等も含む
制約	MPI-2(動的プロセス)対応, NFS(共有ディスク)対応, 子プログラムはMPIでコンパイルされている事。	MDはノード内並列のみ対応 MPIを使わない
特徴	１ノード分、MD制御プロセスとして使用する。レプリカ数 x MDの並列数 + 1ノード使用する。	MD制御プロセスは各々のレプリカの使用するノード上で起動する。
計算速度	新しい機能なのでシステムのmpi_comm_spawnの対応/整備状況に依存。大規模系で大規模計算向き。	速い

MPI_COMM_SPAWN

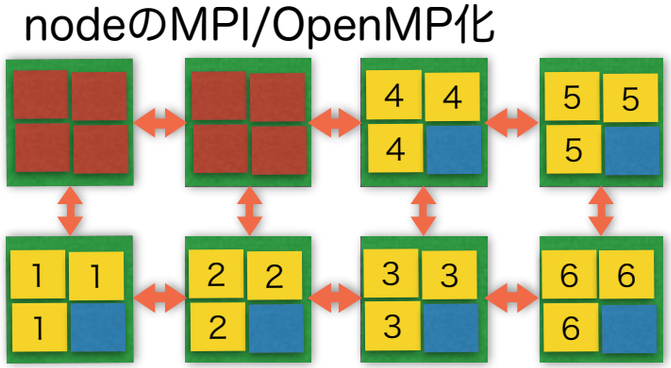
- 動的プロセス生成
  - MPIのプロセスからMPIを実行する
  - MPI-2に対応していればどの計算機でも動く
  - スケジューラが制限をかけている場合がある
  - Master processが必要
  - Child Jobはnodeをまたぐ事ができる

例：4 core x 8 nodeのシステム

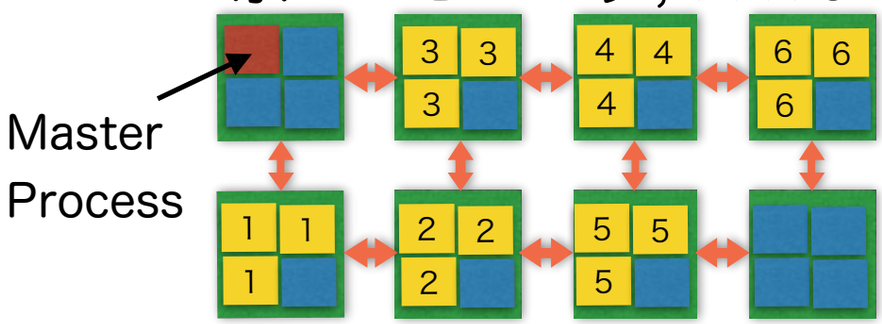
制限の無いPCクラスタ



京コンピュータ, FX10でMaster nodeのMPI/OpenMP化



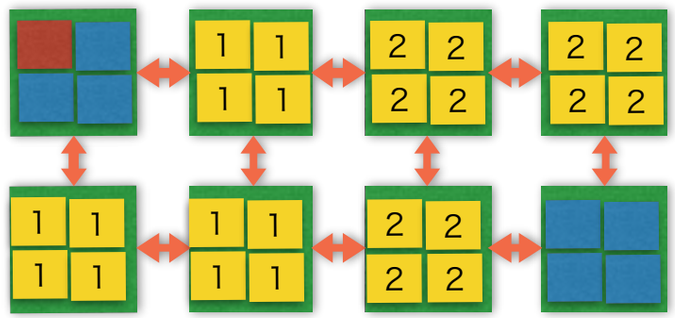
Master process: 1 core
Child Job: 3 core x 6 Job生成
京コンピュータ, FX10



1つのノード内に2つ以上のJobは入らない

Master process: 1 core
Child Job: 12 core x 2 Job生成
京コンピュータ, FX10

Child Jobはノードをまたぐ事ができる

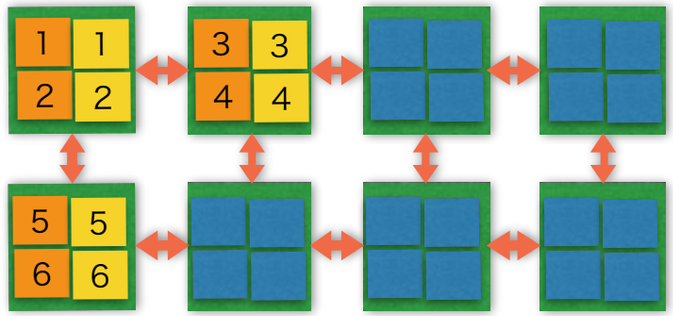


system 関数を用いたレプリカの制御

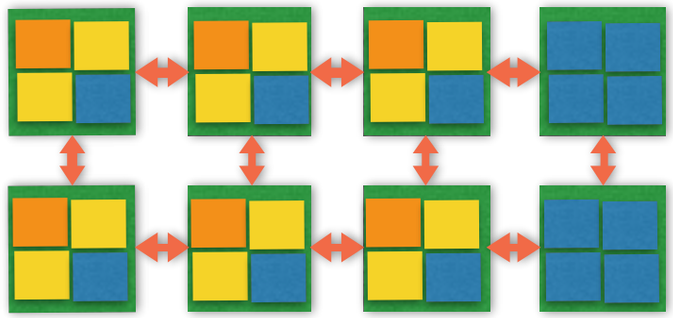
- MPIのプロセスからSystem関数を実行
- 大抵のPCクラスタで動く
- しばしばスケジューラが制限をかけている場合がある
- Master processは必要ないが、Child Jobの実行プロセスの１coreが制御にも使われる。
- 現状ではMPIプロセスを生成できない（nodeをまたぐ事ができない）

例：4 core x 8 nodeのシステム

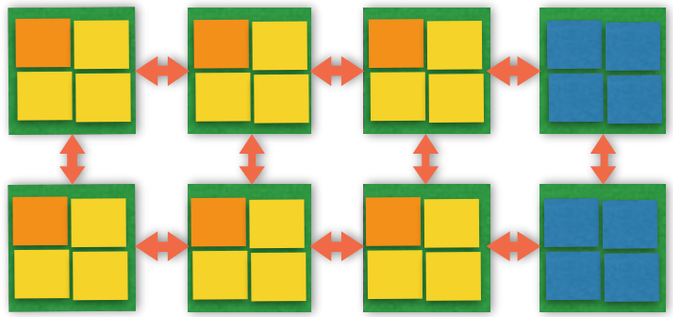
Child Jog: 2 core x 6 Job生成



Child Jog: 3 core x 6 Job生成



Child Jog: 4 core x 6 Job生成



３つのサポーティングソフトウェア

名称	機能	理由
input_builder	REINのインプットを自動生成する	柔軟性の為、数多くのインプットファイルが必要であるがそれらを一括で自動生成する
batch_builder	バッチキュー用のスクリプト生成と、REINのインプットの修正を自動で行なう	レプリカ数やノード数、スレッド数、ステージングの設定などの設定は非常に複雑な為
remd_convert	レプリカ毎やパラメータ毎にトラジェクトリーファイルを収集する	レプリカが多くなると、データ収集が非常に複雑になる為

サポーティングプログラムのしくみ

input_builder　　← ib.inp

$ input_builder -h ctrl　　インプットのテンプレート

$ input_builder -h ctrl **all** 全てのオプションが表示される

$ input_builder ib.inp　　実行

テンプレートファイルを読んで、reinのインプットを作成

- namdやmarble, rein.inpのテンプレートが入っている
- 必要に応じてテンプレートを直す
- 追加も同じ形式（形式）で作成できる

基本的には３ステップで実行

1) **Input_builder** → rein.inp　reinのインプットを自動作成

#input_builder ib.inp

2) **Batch_builder** → rein.sh　ステージングなどの煩雑なバッチスクリプトを自動生成

#batch_builder bb.inp

3) あとはバッチスクリプトを投入するだけ。

pjsub rein.sh

Restart

#batch_builder bb.inp restart　（restartと付けるだけで自動生成）

remd_converter → トラジェクトリをまとめる

#remd_converter rc.inp

インストール

REINパッケージの構成

2014/02/26時点でversion 1.0.4です。

```
REIN----- README.md            : README file
      |-- AUTHORS.md           : authors file
      |-- note.md              : release notes
      |-- update.md      : update
      |-- COPYING              : GPL licence file
      |-- src/                 : main code of REIN
      |-- tools/               : supporting software for REIN
      |-- examples/            : For tutorial
      `-- manual/manual.pdf    : manual
```

情報：　@rein_devel　http://twitter.com/rein_devel

質問など：　yukimya+rein@gmail.com

REINのコンパイル1

京コンピュータ、FX10

```
$ cd src
$ ./configure.sh sparc k
```

京用のNAMDのバイナリは提供しますのでご連絡下さい。

PCクラスタなど(no spawn版)

```
$ cd src
$ ./configure.sh x86_64 gfortran nospawn
```

NAMDのhomepageからDLしたバイナリがそのまま使えます。

PCクラスタなど(spawn版)

```
$ cd src
$ ./configure.sh x86_64 ifort
```

NAMDをMPI用にコンパイルして下さい。

REINのコンパイル２

インストール

```
$ make
$ make install
```

src/と同じ場所にあるbin/にreinという実行ファイルが格納されます

clean

```
$ make clean
```

objectファイルなどが消去されます。

distclean

```
$ make distclean
```

objectファイルだけでなくbin/に入っているreinも消去されます。

サポーティングソフトウェアのコンパイル

京コンピュータ、FX10

```
$ cd tools
$ ./configure x86_64 gfortran
```

京, fx10のフロントエンドはx86_64マシンなので

PCクラスタなど(no spawn版)

```
$ cd src
$ ./configure x86_64 ifort
```

ifortも使えます。gfortranも可。

インストール

```
$ make
$ make install
```

bin/にinput_builder, batch_builder, remd_converter, templates/
がインストールされます。clean, distcleanはreinと同じ。

実行の為の準備

REIN実行前の準備１

事前に、通常の分子動力学シミュレーションを行う時と全く同じ準備をする。

1. NAMD: VMDなどでpdbファイルからpsfファイルを作成。（REINの初期設定の力場はC27+CMAP）
   MARBLE: molxでmdatとcrdファイルを作成。
2. MDプログラムを用いてMinimizationする。
3. MDプログラムを用いて平衡化計算を行う。

REIN実行前の準備２

REMD/MREMシミュレーションのプログラムとインプットの準備をします。

ここではrun/というディレクトリ内でシミュレーションを行うとする。
ここでは、最も単純な方法を述べる。

1. rein/bin/にある全てのソフトウェアとtemplates/をrun/にコピーする。
2. 平衡化で得られた計算結果(coord, vel, xsc, xst)とinputのpdb, psfを、initial.XXX (XXXはそれぞれの拡張子)という名前に変えてrun/へコピーする。
3. parameterファイルもrun/へコピーする

REIN実行前の準備３

input_builderを使って
reinのinputファイルと
実行用バッチファイルを作成する。

1. $ cd run/
2. $ ./input_builder -h ctrl > ib.inp
3. $ vi ib.inp　　レプリカの情報とPMEメッシュの修正

   その他はbatch_builderでも修正可能
4. $ ./input_builder ib.inp > ib.out
5. rep.1/, rep.2/, ...というディレクトリと、file_*.dというファイルが生成。rein.inpのひな形が生成。

例: Input file of Input builder

REINのインプットは複雑な計算にも対応している。
簡単にREINのインプットを自動生成する。

```
# control parameters in input_builder
rein_input            = rein.inp      # rein input file name
spawn  =  yes
[REMD]                  1次元目：この場合、温度
axis001               = temperature   # axis (TEMPERATURE/HARMONIC)
num_replicas001       = 4             # # of replicas for the axis
range001              = 300.0 360.0   # lowest, highest value
division001           = exp           # (EQUIV/EXP/CYCLE)

axis002               = harmonic      2次元目：この場合、原子間距離
num_replicas002       = 4
range002              = 2.0 8.0
division002           = equiv                 等間隔
distance002           = ADP CR 1-1 ADP CL 1-1
spring_const002       = 2.0

[MD]
md_program            = namd          # md program (NAMD/MARBLE)
md_exe_file           = ./namd2       # md exe file

md_temperature        = 300.0         # md temperature
md_steps              = 1000          # md steps / exchange
system_size_x         = 23.4          # system size --automatically create pme grids
system_size_y         = 23.2          #
system_size_z         = 23.9          #
```

システムサイズを入れると自動的にPMEのメッシュの数を計算する

温度 - 距離
2D-REMD

ex)　Alanine Dipeptide

REIN/NAMD, NVT

MD steps: 1000 steps (2ps)

Exchange: 100 exchanges

Number of Replicas:
　Temperature : 4 replicas
　Distance　　: 4 replicas
　Total: 4 x 4 = 16 replicas

Range of Temperature: 300-360K
Range of Distance　　: 2.0-8.0Å

$ input_builder ib.inp
実演

REINのインプット：サポーティングソフトウェアで自動生成

```
sample/ ------ rein.inp                (For REIN)
        |
        |--- file_rank.d               (For REIN)      レプリカとパラメータの対応表
        |--- file_value.d              (For REIN)      パラメータの具体的な値
        |--- file_on.d                 (For REIN)      計算に用いるレプリカ（input_builderのoutputをそのまま使って下さい）
        |--- file_axis_ex.d            (For REIN)    軸の順序（同上）
        |--- file_coef.d               (For REIN)      インプットに対する温度比
        |--- file_ex.d                 (For REIN)      交換ルール
        |
        |--- initial.pdb               (NAMD)              テンプレート構造
        |--- initial.psf               (NAMD)              トポロジーファイル
        |--- par_all27_prot_lipid.prm   (NAMD)           パラメータファイル
        |--- initial.inp               (NAMD modified)           namdインプットファイル。パラメータにタグを付けている。
        |--- initial.colvar  [for REUS] (NAMD modified)      namdインプットファイル。束縛情報。同上
        |
        |--- rep.1/ ----- rep.1.0.coor  (NAMD)             初期構造
        |         |--- rep.1.0.vel   (NAMD)                 初期速度
        |         `--- rep.1.0.xsc   (NAMD)                 初期環境（体積、圧力）
        |--- rep.2/ ----- rep.2.0.coor  (NAMD)
        |         |--- rep.2.0.vel   (NAMD)
        |         `--- rep.2.0.xsc   (NAMD)
        |--- rep.3/ ----- rep.3.0.coor  (NAMD)
        |         |--- rep.3.0.vel   (NAMD)
        |         `--- rep.3.0.xsc   (NAMD)
        `--- rep.4/ ----- rep.4.0.coor  (NAMD)
                  |--- rep.4.0.vel   (NAMD)
                  `--- rep.4.0.xsc   (NAMD)
```

自動生成して、必要な部分を修正すると良い

REIN実行前の準備４

batch_builderを使って
計算機環境に合わせたreinのinputの書き換え、
と実行用バッチファイルを作成する。

1. $ ./batch_builder -h ctrl > bb.inp
2. $ vi bb.inp　MPIやthreadに関する情報を記載する。
　　スケジューラに関する情報を記載する。
3. $ ./batch_builder bb.inp > bb.out
4. rein.inpが完成し、rein.shが生成

例: input file of Batch builder

レプリカの数だけ、ファイル転送する必要があるが、
その際のステージングやFTLなど面倒な記述を自動で作成する。

```
# control parameters in batch_builder

rein_input              = rein.inp       # REIN input file name
spawn                   = yes
chain_job               = no
[BATCH]
batchfile_name          = rein.sh        # batch filename
job_scheduler           = K              # job scheduler type ([K]/GE/RICC)
queue_name              = small       # queue name, k:([SMALL]/LARGE),fx10: REGULAR..

num_mpiproc_rein        = 2              # number of process for rein
num_threads_md          = 8              # number of threads in md
num_mpiproc_md          = 4              # # of mpi process for md
cpu_time                = 01:00:00       # cpu time
num_core_cpu            = 8    # number of core in cpu, for k = 8, for fx10 = 16, ...

[REMD]
replica_exchange        = yes  # ([YES]/NO): replica exchange / umbrella sampling
num_of_exchange_steps   = 100              # number of exchange steps
```

REIN: 16 core (2 node)

MD: 4 node x 16 replica = 64 node
(512 core)

Total node: 66 node

Thread: 8 (hybrid MPI)

Exchange steps :
100 exchanges

現在、K (fx10) , RICC，SGE タイプのスケジューラに対応.
新規スケジューラもテンプレートの追加で対応できる.

$ batch_builder bb.inp
$ batch_builder bb.inp restart

REINの実行

REIN実行

1. $ pjsub rein.sh
2. $ ./batch_builder bb.inp restart > bb.out2

REIN終了後、run/内に結果データの入ったSTGOUT_DIR/が作成される。
batch_builderでrestartをかけると全ての結果ファイルがresults/の中に格納される。

以上を繰返す。

実行後：トラジェクトリーをreplica単位かパラメータ単位にまとめ直す

1. $ ./remd_convert -h ctrl > rc.inp
2. $ vi rc.inp　　ファイルのあるディレクトリを指定
3. $ ./remd_convert rc.inp > rc.out

例: input file of remd_convert

解析の前に、結果のトラジェクトリをレプリカ毎かパラメータ毎にまとめると便利

```
# control parameters in remd_convert

[INPUT]

psffile           = initial.psf
reffile           = initial.pdb
trj_format        = DCD          # (DCD/MARBLE/PDB)

directory001      = ./results/result0000001_0000010
directory002      = ./results/result0000011_0000020
directory003      = ./results/result0000021_0000030

repeat001         = 3
trj_filename001   = rep.{rep_no}/rep.{rep_no}.{step}.dcd
energy_trj001     = replica.trj

energy_column     =         # column of output energy, ex. = 1 2-4, empty: all column

[OUTPUT]

out_pdbfile       = out.pdb
trj_filename      = out.{type}.{number}.{extension}
trj_format        = DCD          # (DCD/MARBLE/AMBER/TINKER/PDB/GROMACS)

[CONVERT]

type              = REPLICA      # (REPLICA/CONDITION)
rank              =              #  ex. = 1 2-8,  empty: output all rank

[SELECTION]

# select_atom       = all
# select_atom       = molname:protein | molname:lipid
# mole_name001      = protein  P1:1:TYR P1:5:MET
# mole_name002      = lipid    OLEO:PCGL:OLEO
```

REINを3回リスタートした場合の例

レプリカでまとめている。

$ remd_convert rc.inp

REIN連続投入

京、FX10の場合：tools/auto_submitter/auto_submitter.plを用いる。

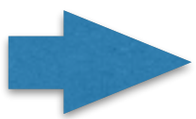
- $ vi auto_submitter.pl
- $ perl auto_submitter.pl

PCクラスタの場合、２通り

- auto_submitter.plの利用
- chain_job = yes (bb.inp)とすると
  rein.shの最後に１行加わる。

```
...
${MPIEXEC} ${EXEC}

./batch_builder bb.inp restart
```



```
...
${MPIEXEC} ${EXEC}
./batch_builder bb.inp restart
${MPIEXEC} ${EXEC}
./batch_builder bb.inp restart
```

Output format (replicat.trj)

```
#========= EXCHANGE_STEP = 1
       1       1             290.14150          -1624.33470          370.15570          -1994.49040
       2       2             309.04970          -1677.37840          394.27840          -2071.65680
       3       3             314.56290          -1658.91060          401.31190          -2060.22250
       4       4             313.83160          -1657.91540          400.37900          -2058.29450
       5       5             304.91160          -1642.47860          388.99900          -2031.47760
       6       6             309.84760          -1635.80170          395.29630          -2031.09800
       7       7             299.44280          -1685.76920          382.02210          -2067.79130
       8       8             288.62440          -1683.47140          368.22020          -2051.69170
 #========= EXCHANGE_STEP = 2
       1       2             306.76840          -1664.28540          391.36800          -2055.65340
       2       1             317.18100          -1651.98420          404.65210          -2056.63630
       3       4             310.83750          -1629.92330          396.55920          -2026.48240
       4       3             284.34980          -1655.03960          362.76680          -2017.80640
       5       6             294.21380          -1680.86470          375.35110          -2056.21580
       6       5             336.64260          -1606.88210          429.48070          -2036.36280
       7       8             296.55280          -1698.13390          378.33510          -2076.46900
       8       7             299.56080          -1662.84080          382.17260          -2045.01340
```

Replica#, condition#, Temperature, (Distance etc..), Potential, Kinetic,Total, Extra

```
#========= EXCHANGE_STEP = 11
    1    1        322.72880      6.61182     -1643.34560      411.72990     -2055.07550      6.52260
    2    8        333.08190      7.13212     -1575.02380      424.93810     -1999.96190      7.48020
    3    6        355.33950      6.55514     -1523.72050      453.33390     -1977.05440      0.09900
    4    5        321.22060      6.67915     -1597.99230      409.80570     -2007.79800      0.05150
    5    2        351.72040      6.10475     -1554.42320      448.71660     -2003.13980      4.81970
    6    9        371.11030      7.01426     -1485.12630      473.45380     -1958.58020      7.94310
    7    7        284.53280      7.15645     -1680.79330      363.00030     -2043.79360      7.38640
    8    4        276.94870      6.58486     -1677.11400      353.32460     -2030.43860      0.08620
    9    3        334.14340      6.26521     -1528.19240      426.29240     -1954.48480      5.33080
#========= EXCHANGE_STEP = 12
```

remd_convertで得られたOutputのformat (out.xxx.#.enetrj)

```
  1       1       1          297.75270       -1671.67350        379.86590       -2051.53940
  2       1       2          317.02580       -1639.88920        404.45410       -2044.34340
  3       1       3          310.59180       -1617.55910        396.24580       -2013.80490
  4       1       4          315.72080       -1601.63910        402.78920       -2004.42820
  5       1       4          319.46520       -1626.64460        407.56620       -2034.21090
  6       1       3          321.01360       -1633.93310        409.54160       -2043.47480
  7       1       3          300.87790       -1678.91540        383.85290       -2062.76830
  8       1       4          298.17360       -1661.42740        380.40290       -2041.83030
  9       1       4          299.28800       -1656.02570        381.82470       -2037.85040
 10       1       4          295.37950       -1657.20690        376.83830       -2034.04520
```

#, Replica#, condition#, Temperature, (Distance etc..), Potential, Kinetic,Total, Extra

```
 1       1       1          286.98240        6.37033       -1717.17430        366.12540       -2083.29970        5.67950
 2       1       1          312.47330        6.44694       -1653.73810        398.64610       -2052.38420        5.94070
 3       1       1          289.37010        6.00928       -1643.07150        369.17160       -2012.24300        4.52790
 4       1       1          296.36820        6.23849       -1679.83210        378.09960       -2057.93170        5.24390
 5       1       1          289.88810        6.28432       -1653.50920        369.83240       -2023.34170        5.39340
 6       1       2          314.75100        6.02690       -1618.84470        401.55200       -2020.39670        4.58110
 7       1       2          330.75970        6.42143       -1606.04790        421.97540       -2028.02330        5.85310
 8       1       2          332.70610        6.61133       -1573.92340        424.45860       -1998.38200        6.52090
 9       1       2          330.66870        5.60459       -1552.42150        421.85930       -1974.28080        3.39190
10       1       1          308.05460        6.21432       -1668.87810        393.00890       -2061.88700        5.16590
```

まとめ

レプリカ交換インターフェースプログラム
Replica-exchange interface program for K
REIN-K

プログラム開発の目的	京コンピュータで大規模な多次元レプリカ交換シミュレーションを実施する事を想定したプログラム 様々なMDプログラムを利用したレプリカ交換をサポート
ファンド	次世代計算科学研究開発プログラム　次世代生命体統合シミュレーション研究開発プロジェクト(ISLIM)にて作成
その他ファンド	理化学研究所生命システム研究センター(QBiC)
講習会 ダウンロードサイトなど	SCLS（HPCI戦略プログラム 戦略分野1） RIKEN情報基盤センター バイオグリッドセンター関西、バイオスーパーコンピューティング研究会

REINの情報

Main developer	Naoyuki Miyashita
Author	Naoyuki Miyashita and Yuji Sugita
ライセンス	ライセンス：GPL version 3
引用	Naoyuki Miyashita, Yuji Sugita, M-2: Replica-exchange interface program (REIN), http://www.islim.org/islim-dl_e.html
サポート	理研QBiC 宮下
e-mail	yukimya+rein@gmail.com
Twitter	@rein_devel　http://twitter.com/rein_devel

謝辞

- 杉田有治 主任研究員, 理研, 理研QBiC, 理研AICS
- 池口満徳 准教授, 横浜市大
- 李秀栄 研究員, 理研
- 二島渉 協力研究員, 理研, Pai-chi Li 協力研究員, 理研
- 高瀬規男, 磯子ソフト
- HPCIヘルプデスク
- 松田元彦 研究員, 理研AICS, 丸山直也 チームリーダー, 理研AICS
- 本研究開発は以下の支援を受けて行われたものである。
  - 文部科学省　最先端・高性能汎用スーパーコンピューターの開発利用「次世代生命体統合シミュレーションソフトウェアの研究開発」(RIKEN SCRP ISLiM)
  - 理研　生命システム研究センター
  - 理研　計算科学研究機構　京コンピュータ
  - JSPS科研費　若手研究（B）　Number 24700299.
  - 理研　計算機センター RIKEN Integrated Cluster of Clusters (RICC)

実習手順書

各自のPCにダウンロード

http://www.islim.org/islim-dl_j.html

ライフサイエンスのグランドチャレンジ・アプリケーション・プロジェクト

ISLiM

次世代生命体統合シミュレーションソフトウェアの研究開発

Next-Generation Integrated Simulation of Living Matter

English

HOME | 概要 | 研究チーム紹介 | 開発アプリケーション | 行事・研究会 | 情報ライブラリ | 問合せ先 | リンク

ダウンロード

(Updated on 2013/3/28)

●ダウンロードについて

本プロジェクトではその成果を社会に還元する一環として、研究開発の成果物である、スーパーコンピュータ「京」で稼働する開発ソフトウェアをプロジェクト期間中(2006年10月～2013年3月末)にソースコード・レベルで順次公開していきます。ただし一部の開発ソフトウェアについてはソースコードの公開ができないため、バイナリー・コードによる公開となります。 また中間成果物である、高並列クラスター上で稼働するバージョンも同様に可能な範囲で公開いたします。 産業界での利用、あるいは学生・大学院生の教育への活用など、広くご利用ください。 (本プロジェクトでは複数の大学、研究機関から研究開発者が参加・連携してプロジェクトを進めています。そのため、研究開発者の所属機関のダウンロードサイトにリンクしているケースもありますので、ご了承ください。またプロジェクトの進捗に合わせ、ダウンロードについての内容も更新されますので、最新情報についてはこのページでご確認ください。)

分子スケール・アプリケーション

量子科学計算(QM)、分子動力学計算(MM)、粗視化モデル計算(CG)の手法を結合したQM/MM、MM/CG法によって、マルチスケールシミュレーションを実現するためのプログラム。

クリック

番号	アプリケーション	コード名	ダウンロード	開発機関
M-1	マルチコピー・マルチスケール分子シミュレーション法開発の基盤となるクラスライブラリ	mu2lib-K (開発コード名 Platypus-MM/CG)	ダウンロード	理化学研究所
M-2	レプリカ交換分子動力学計算インターフェイス	REIN-K (開発コード名 Platypus-REIN)	ダウンロード	理化学研究所
M-3	全原子分子動力学計算	MARBLE-K	ダウンロード	横浜市立大学
M-4	粗視化モデル計算	CafeMol-K	ダウンロード	京都大学

ダウンロード

REIN
Replica-exchange interface

● REIN-K (開発コード名: plutypus-rein) version 1.0.4
レプリカ交換インターフェースプログラム (Replica-Exchange INterface Program: REIN)
License: GPLv3

REIN(開発コード名:plutypus-rein)は、多次元レプリカ交換分子動力学計算を、NAMDなどの既存のMDプログラムを用いて行うインターフェースプログラムです。タンパク質などの生体分子の構造予測や、複数の生体分子の配置予測、２つの分子の結合自由エネルギープロファイル計算などを目的としています。

このversionではMDはNAMDのみに対応しております。

また、本プログラムは京コンピュータで動作しますが、コンパイルオプションを変更する事で、FX10やPCクラスターにも対応できる様になっています。 プログラムに関する最新情報はtwitterで配信いたします。[rein_devel] (https://twitter.com/rein_devel)をfollowして下さい。

質問なども受け付けますが、込み入った内容はメール yukimya+rein [a] gmail.com　にお願い致します。([a]は@に置き換えて下さい。)

下記にご記入の上送信ボタンを押して下さい。

お名前 (必須)	名前
所属 (必須)	所属
部門名	部門名
職業 (必須)	◯研究者・開発者 ◯学生・院生 ◯その他
E-Mail	your@e-mail your@e-mail (確認)
REIN-Kについての情報をEmailで受取ることを了承しますか	◯了承します。 ◯必要ありません。

選択

選択

クリック

送信　リセット

入力された個人情報の扱いは、理化学研究所の個人情報保護規程に準拠します。

rein-v.1.0.4.tarをscls fx10へコピー

```
$ scp rein-v.1.0.4.tar.gz hoge@hpci-scls.riken.jp:.
```

rein-v.1.0.4.tarを展開

```
$ tar -zxvf rein-v.1.0.4.tar.gz
$ cd rein-v.1.0.4
```

```
[miya@scls rein-v.1.0.4]$ ls
AUTHORS.md  COPYING  README_v.1.0.4.md  examples  manual  notes_v1.0.4.md  src  tools  update_v.1.0.4.md
[miya@scls rein-v.1.0.4]$
```

reinのコンパイル

```
$ cd src
```

```
[miya@scls src]$ ls
Makefile  Makefile.lib  arch  configure.sh  lib  libmarble  libnamd  main
[miya@scls src]$
```

reinのコンパイル続き

$ sh configure.sh sparc k.openmp

```
[miya@scls src]$ ls
Makefile  Makefile.comp  Makefile.lib  Makefile.machine  arch  configure.sh  lib  libmarble  libnamd  main
[miya@scls src]$
```

$ make

```
[miya@scls src]$ ls
Makefile       Makefile.lib      arch          lib        libnamd  rein_k_openmp
Makefile.comp  Makefile.machine  configure.sh  libmarble  main
[miya@scls src]$
```

$ make install

$ cd ..

```
[miya@scls rein-v.1.0.4]$ ls
AUTHORS.md  COPYING  README_v.1.0.4.md  bin  examples  manual  notes_v1.0.4.md  src  tools  update_v.1.0.4.md
[miya@scls rein-v.1.0.4]$
```

$ ls bin

```
[miya@scls rein-v.1.0.4]$ ls bin
rein
[miya@scls rein-v.1.0.4]$
```

サポーティングソフトウエアのコンパイル

$ cd tools

```
[miya@scls tools]$ ls
Makefile  arch  auto_submitter  batch_builder  configure.sh  input_builder  lib  remd_converter
[miya@scls tools]$
```

$ sh configure.sh x86_64 gfortran

```
[miya@scls tools]$ sh configure.sh x86_64 gfortran
[miya@scls tools]$ ls
Makefile  Makefile.machine  arch  auto_submitter  batch_builder  configure.sh  input_builder  lib  remd_converter
[miya@scls tools]$
```

$ make

$ make install

$ cd ..

$ ls bin

```
[miya@scls rein-v.1.0.4]$ ls bin
batch_builder  input_builder  rein  remd_convert  templates
[miya@scls rein-v.1.0.4]$
```

全てrein-v.1.0.4/binに格納される

templatesには様々な設定のテンプレートが入っている

```
[miya@scls rein-v.1.0.4]$ ls bin/templates/
t_archive.sh      t_batch_ricctar     t_namd.inp          t_rein_axis.inp     t_restart_ricc_upc
t_batch_fx10      t_marble.inp        t_namd_colvar_ang   t_reininp_modify    t_restart_ricctar
t_batch_ge        t_marble_atom_h     t_namd_colvar_dih   t_restart_ge        t_staging_k_marble
t_batch_k         t_marble_copyfiles  t_namd_colvar_dis   t_restart_k         t_staging_k_namd
t_batch_nostfx10  t_marble_group      t_namd_copyfiles    t_restart_nostfx10  t_staging_ricc_namd
t_batch_ricc      t_marble_group_h    t_namd_harmonic     t_restart_num       t_staging_ricctar_namd
t_batch_ricc_upc  t_namd.colvar       t_rein.inp          t_restart_ricc
[miya@scls rein-v.1.0.4]$
```

- t_batch : バッチキューのテンプレート
- t_marble, t_namdなど : それぞれのMDのインプットのテンプレート
- t_rein : reinの基本設定
- t_restart : リスタート時の動作ルール
- t_staging : スパコンなどで、ファイル転送がある場合の転送ルールを記載

例題を試す

examplesにalanine dipeptideの例が格納されている

remd:　温度レプリカ交換

reus:　end-to-end距離のレプリカ交換

mrem: 温度とend-to-end距離の二次元のレプリカ交換

シミュレーションの準備

$ cd examples/remd/namd

```
[miya@scls namd]$ ls
bb.inp           bb.inp.ricctar  ib.inp.nostfx10  initial.coor  initial.psf  initial.xsc                rein
bb.inp.nostfx10  ib.inp          ib.inp.ricc      initial.pdb   initial.vel  par_all27_prot_lipid.prm
[miya@scls namd]$
```

ツールを全てコピーする

$ cp -r ../../../bin/* .　　#必要なプログラムをコピー

$ cp /home/miya/bin/namd2 .　　# namd2をコピー

```
[miya@scls namd]$ ls
batch_builder    bb.inp.ricctar   ib.inp.ricc   initial.psf  input_builder              rein
bb.inp           ib.inp           initial.coor  initial.vel  namd2                      remd_convert
bb.inp.nostfx10  ib.inp.nostfx10  initial.pdb   initial.xsc  par_all27_prot_lipid.prm  templates
[miya@scls namd]$
```

Input builderによるreinのインプット作成

$ ./input_builder ib.inp.nostfx10

```
[miya@scls namd]$ ls
batch_builder    file_coef.d   ib.inp           initial.pdb    namd2                        rep.1  rep.6
bb.inp           file_ex.d     ib.inp.nostfx10  initial.psf    par_all27_prot_lipid.prm     rep.2  rep.7
bb.inp.nostfx10  file_on.d     ib.inp.ricc      initial.vel    rein                         rep.3  rep.8
bb.inp.ricctar   file_rank.d   initial.coor     initial.xsc    rein.inp                     rep.4  templates
file_axis_ex.d   file_value.d  initial.inp      input_builder  remd_convert                 rep.5
```

それぞれのレプリカのディレクトリ(rep.#)とreinのインプットファイル(file_xx.d, rein.inp)、MDのテンプレート(initial.inp)が自動生成される。

batch builderによるreinのインプット作成

$ ./batch_builder bb.inp.nostfx10

```
[miya@scls namd]$ ls
batch_builder    file_coef.d   ib.inp           initial.pdb    namd2                        remd_convert  rep.5
bb.inp           file_ex.d     ib.inp.nostfx10  initial.psf    par_all27_prot_lipid.prm     rep.1         rep.6
bb.inp.nostfx10  file_on.d     ib.inp.ricc      initial.vel    rein                         rep.2         rep.7
bb.inp.ricctar   file_rank.d   initial.coor     initial.xsc    rein.inp                     rep.3         rep.8
file_axis_ex.d   file_value.d  initial.inp      input_builder  rein.sh                      rep.4         templates
[miya@scls namd]$
```

reinの実行

$ pjsub rein.sh

jobの状態チェック

$ pjstat

```
[miya@scls namd]$ pjstat

  ACCEPT QUEUED  STGIN  READY RUNING RUNOUT STGOUT   HOLD  ERROR   TOTAL
       0      0      0      0      1      0      0      0      0       1
s      0      0      0      0      1      0      0      0      0       1

JOB_ID     JOB_NAME   MD ST  USER     START_DATE      ELAPSE_LIM NODE_REQUIRE
14306      rmd2       NM RUN miya     02/25 15:45:52  0000:20:00 10
```

$ pjstat -A

jobの出力状態チェック

$ pjcat -o #jobnumber

$ pjstat -e #jobnumber

```
[miya@scls namd]$ pjstat

  ACCEPT QUEUED  STGIN  READY RUNING RUNOUT STGOUT   HOLD  ERROR   TOTAL
       0      0      0      0      0      0      0      0      0       0
s      0      0      0      0      0      0      0      0      0       0

[miya@scls namd]$ ls
batch_builder    file_ex.d        ib.inp           initial.psf              rein          rep.3  replica.trj
bb.inp           file_on.d        ib.inp.nostfx10  initial.vel              rein.inp      rep.4  rmd2.e14306
bb.inp.nostfx10  file_out_coef.d  ib.inp.ricc      initial.xsc              rein.sh       rep.5  rmd2.i14306
bb.inp.ricctar   file_out_rank.d  initial.coor     input_builder            remd_convert  rep.6  rmd2.o14306
file_axis_ex.d   file_rank.d      initial.inp      namd2                    rep.1         rep.7  templates
file_coef.d      file_value.d     initial.pdb      par_all27_prot_lipid.prm rep.2         rep.8
[miya@scls namd]$
```

batch builderによるデータ処理と、次のインプット作成

$ ./batch_builder bb.inp.nostfx10 restart

```
[miya@scls namd]$ ls
batch_builder    file_ex.d        ib.inp.ricc   initial.xsc              rein.sh       rep.5          rmd2.i14306
bb.inp           file_on.d        initial.coor  input_builder            remd_convert  rep.6          rmd2.o14306
bb.inp.nostfx10  file_rank.d      initial.inp   namd2                    rep.1         rep.7          templates
bb.inp.ricctar   file_value.d     initial.pdb   par_all27_prot_lipid.prm rep.2         rep.8
file_axis_ex.d   ib.inp           initial.psf   rein                     rep.3         results
file_coef.d      ib.inp.nostfx10  initial.vel   rein.inp                 rep.4         rmd2.e14306
[miya@scls namd]$

[miya@scls namd]$ ls results/
result0000001_0000010
[miya@scls namd]$ ls results/result0000001_0000010/
file_axis_ex.d  file_ex.d  file_out_coef.d  file_rank.d   rein.inp  rep.1  rep.3  rep.5  rep.7  replica.trj
file_coef.d     file_on.d  file_out_rank.d  file_value.d  rein.sh   rep.2  rep.4  rep.6  rep.8
[miya@scls namd]$
```

再度reinの実行

$ pjsub rein.sh

トラジェクトリの整理

$ ./remd_convert -h ctrl > rc.inp

rc.inpファイルの変更

```
# control parameters in remd_convert

[INPUT]

psffile           = initial.psf
reffile           = initial.pdb
trj_format        = DCD          # (DCD/MARBLE/PDB)

directory001      = ./results/result0000001_0000010
directory002      = ./results/result0000011_0000020
directory003      = ./results/result0000021_0000030

repeat001         = 3    (→ 1に変更)
trj_filename001   = rep.{rep_no}/rep.{rep_no}.{step}.dcd
energy_trj001     = replica.trj

energy_column     =              # column of output energy, ex. = 1 2-4, empty: all column

[OUTPUT]

out_pdbfile       = out.pdb
trj_filename      = out.{type}.{number}.{extension}.dcd    (追加)
trj_format        = DCD          # (DCD/MARBLE/AMBER/TINKER/PDB/GROMACS)

[CONVERT]

type              = REPLICA      # (REPLICA/CONDITION)
rank              =              #  ex. = 1 2-8,  empty: output all rank

[SELECTION]
```

remd_convertの実行

$ ./remd_convert rc.inp

```
[miya@scls namd]$ ls
batch_builder    ib.inp           namd2                 out.replica.5.crdtrj      rein          rep.7
bb.inp           ib.inp.nostfx10  out.pdb               out.replica.5.enetrj      rein.inp      rep.8
bb.inp.nostfx10  ib.inp.ricc      out.replica.1.crdtrj  out.replica.6.crdtrj      rein.sh       results
bb.inp.ricctar   initial.coor     out.replica.1.enetrj  out.replica.6.enetrj      remd_convert  rmd2.e14306
file_axis_ex.d   initial.inp      out.replica.2.crdtrj  out.replica.7.crdtrj      rep.1         rmd2.i14306
file_coef.d      initial.pdb      out.replica.2.enetrj  out.replica.7.enetrj      rep.2         rmd2.o14306
file_ex.d        initial.psf      out.replica.3.crdtrj  out.replica.8.crdtrj      rep.3         templates
file_on.d        initial.vel      out.replica.3.enetrj  out.replica.8.enetrj      rep.4
file_rank.d      initial.xsc      out.replica.4.crdtrj  par_all27_prot_lipid.prm  rep.5
file_value.d     input_builder    out.replica.4.enetrj  rc.inp                    rep.6
[miya@scls namd]$
```

out.replica.#.crdtrj : replicaのdcd file
out.replica.#.enetrj : energyなどのトラジェクトリ
番号はレプリカ。更に、[CONVERT] typeを”REPLICA”から”CONDITION”に変更するとコンディションに対するトラジェクトリが得られれる。これらのデータから様々な解析ができる。

是非、VMDなどの画像ソフトウェアを使って動画などを見て下さい。

PCでの簡単な実行例

no spawn option
3 レプリカの場合

インストール

```
$ cd src
$ ./configure.sh x86_64 gfortran local
$ make; make install
$ cd ../tools
$ ./configure.sh x86_64 gfortran
$ make; make install
```

必要なファイルのコピー

```
$ cd ../../test
$ cp -r ../rein/bin/* .
$ cp ../rein/example/input/namd/*
```

予めバイナリ形式のNAMDをnamd ディレクトリへダウンロード

```
$ cp ../namd/Linux_x86_64/namd2 .
$ cp ../namd/Linux_x86_64/charmrun .
```

```
<ib.inp>
rein_input          = rein.inp      # rein input file name
spawn    = no
openmp_rein = yes

$ ./input_builder ib.inp > ib.out

<bb.inp>
rein_input          = rein.inp      # REIN input file name
spawn = no
openmp_rein = yes
[BATCH]
batchfile_name        = rein.sh
job_scheduler         = GE
queue_name            = all.q

num_threads_md        = 1
num_mpiproc_md        = 1
cpu_time              = 01:00:00
num_core_cpu          = 1
num_mpiproc_rein      = 3

$ ./batch_builder bb.inp > bb.out

$ sh ./rein.sh > rein.out1 &
$ ./batch_builder bb.inp restart > bb.out2
$ sh ./rein.sh > rein.out2 &
```