創薬支援アプリ(ISLiMソフトウェア)講習会　２０１５年9月1日
理化学研究所　東京事務所,　　公益財団法人都市活力研究所

REIN-K講習会

宮下　尚之
近畿大学　生物理工学部

理化学研究所　生命システム研究センター

理化学研究所　計算科学研究機構

目次

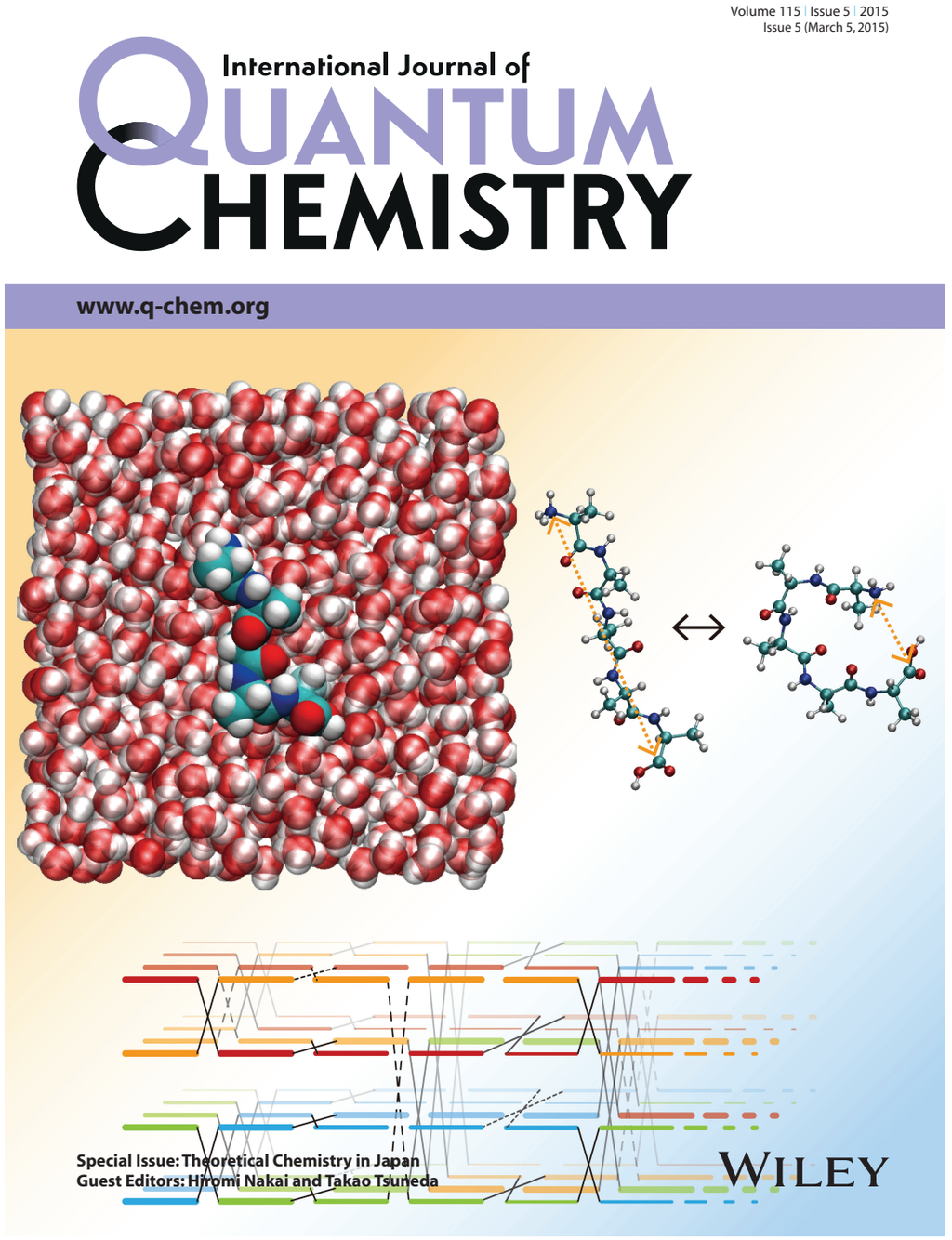
1. レプリカ交換分子動力学法の基礎

2. REIN-Kプログラムの原理

3. レプリカ交換分子動力学法を使った実際の研究例

4. REIN-Kの使い方

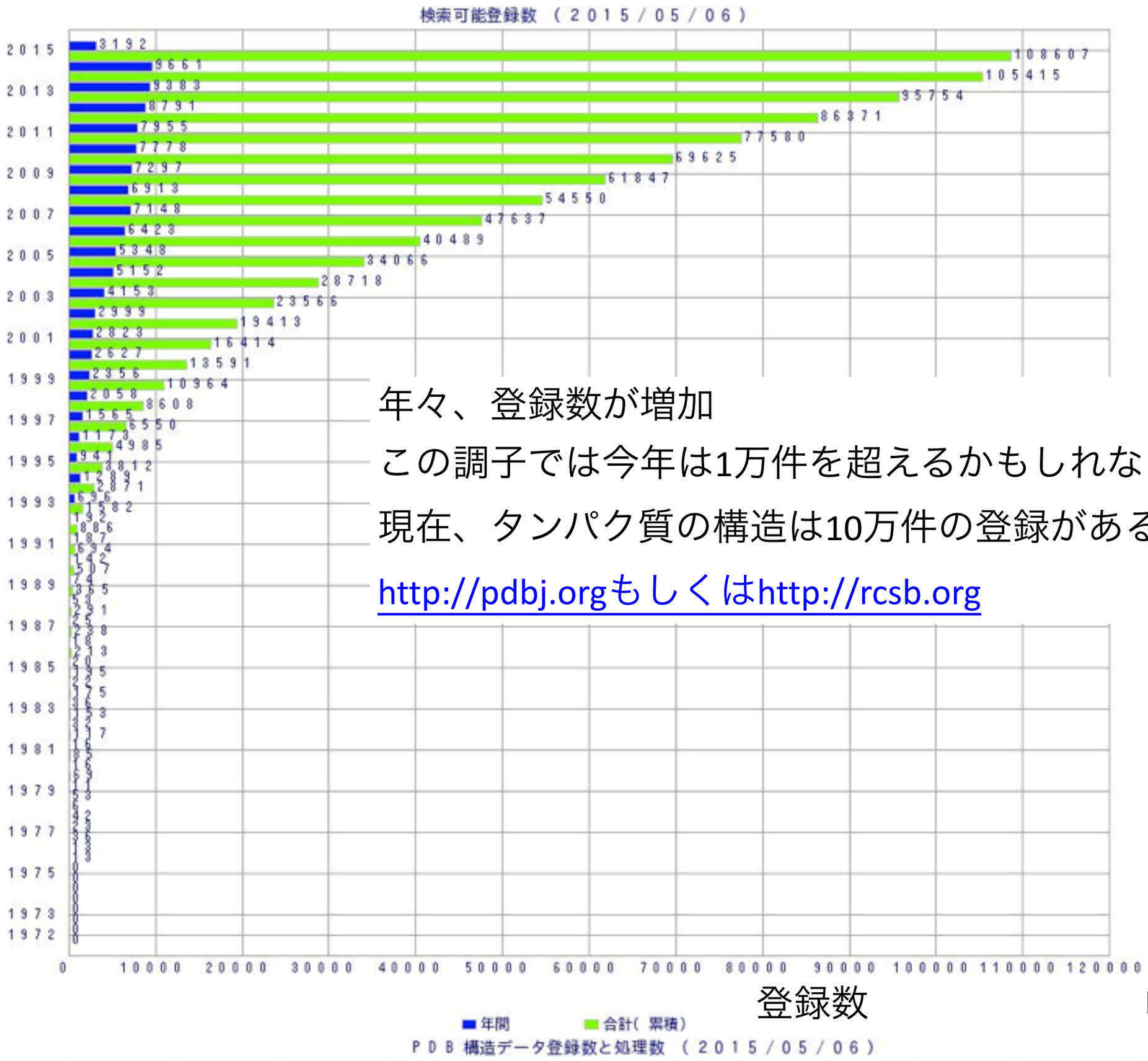
Naoyuki Miyashita, Suyong Re, and Yuji Sugita,
"REIN: Replica-Exchange INterface for simulating protein dynamics and function."
International Journal of Quantum Chemistry,
115(5), 325–332. (2015).
http://doi.org/10.1002/qua.24785



内表紙ですが

1. レプリカ交換分子動力学法の基礎

タンパク質の立体構造はデータバンク(PDB)に登録されている



年

年々、登録数が増加

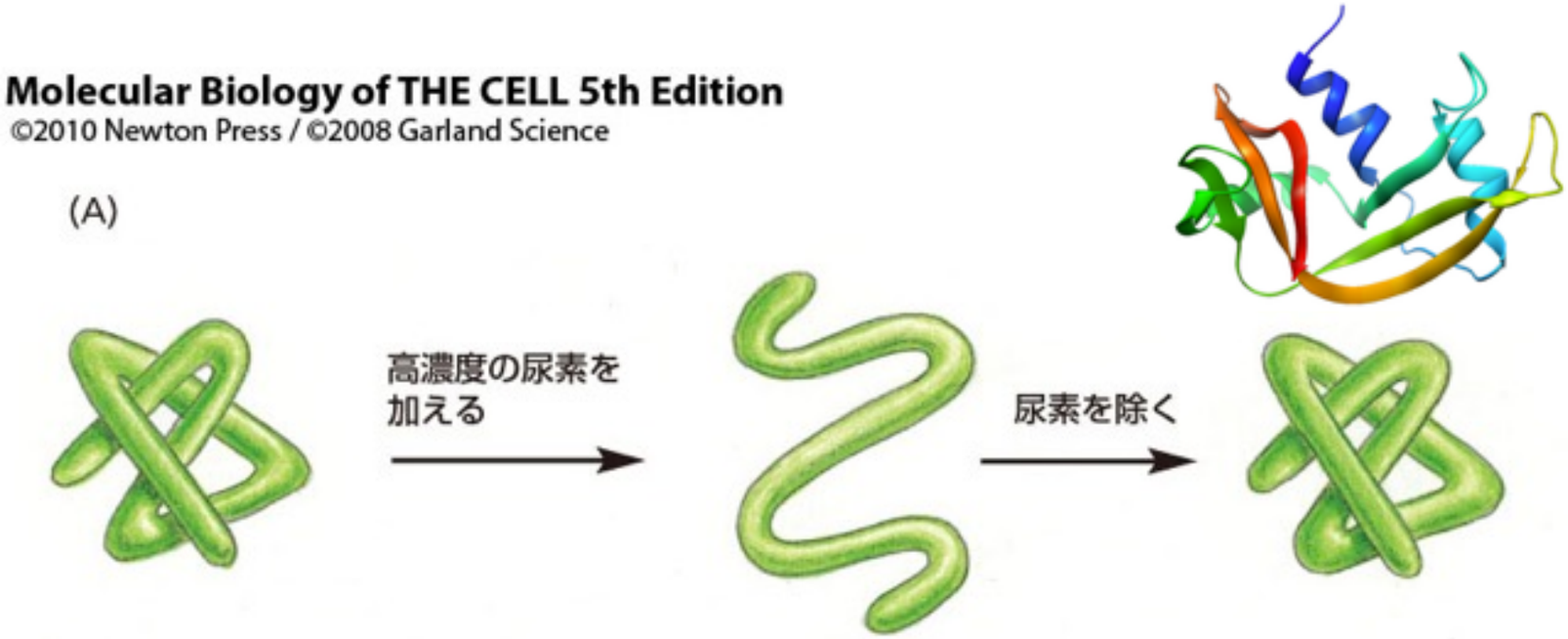
この調子では今年は1万件を超えるかもしれない

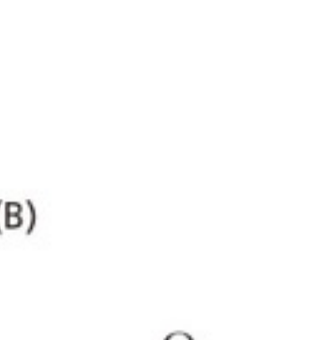
現在、タンパク質の構造は10万件の登録がある。

http://pdbj.orgもしくはhttp://rcsb.org

PDBJより

タンパク質の立体構造（形）の基本　アンフィンセンのドグマ





wikipediaより

1972年ノーベル化学賞
リボヌクレアーゼ分子の研究

リボヌクレアーゼAの実験 ：　一度、変性剤（尿素）で立体構造を壊した後

⇒　変性剤を取り除く（天然条件）⇒　タンパク質の機能が復活する事を発見

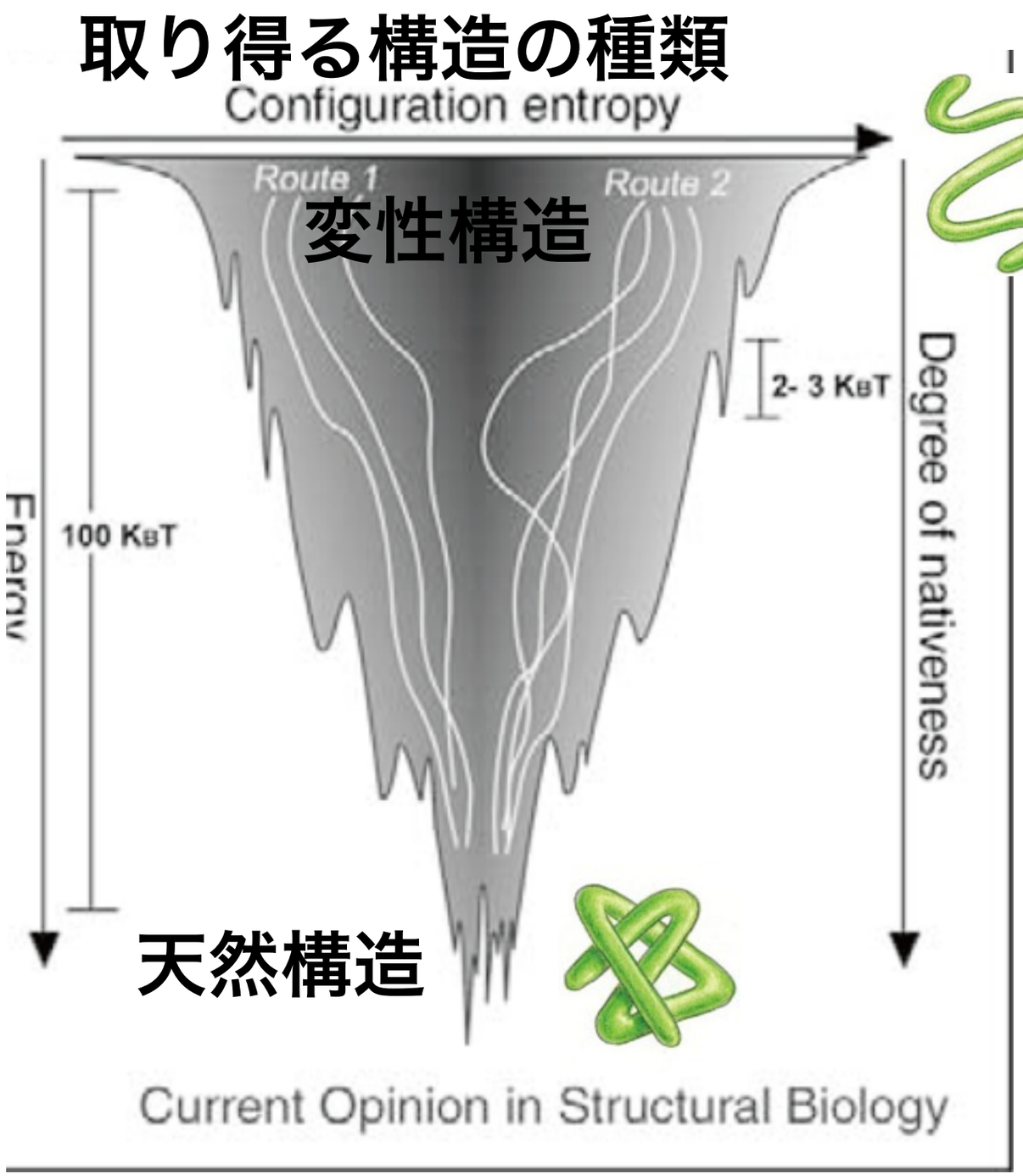
タンパク質の**天然立体構造**はそのアミノ酸の**１次配列**により一意的に**決定**づけられる

タンパク質の天然構造は全系におけるギブス自由
エネルギーが最小の構造に対応する

自由エネルギー的に安定

ファネルモデル　(Peter Wolyness)

タンパク質の
フォールディング



天然構造を見つける
＝最適化問題

タンパク質の天然構造は全系におけるギブス自由
エネルギーが最小の構造に対応する

自由エネルギー的に安定

最適化問題・探索問題

- タンパク質の構造予測や、リガンドータンパク質間の結合予測、構造変化予測などがしたい

最適解を見つける探索問題

- 反応座標に対する自由エネルギーを描きたい

自由エネルギー計算

探索空間が広大　⇒　サンプリング法

サンプリング

- 意味：標本抽出
- ある母集団（調査対象全体）からランダムに標本抽出すること
- →　全体のごく一部を調べるだけで、大きな母集団の正確な情報がつかめる。
- 非常に便利な方法

サンプリング法
レプリカ交換分子動力学法

- 分子動力学法

+

- モンテカルロ法

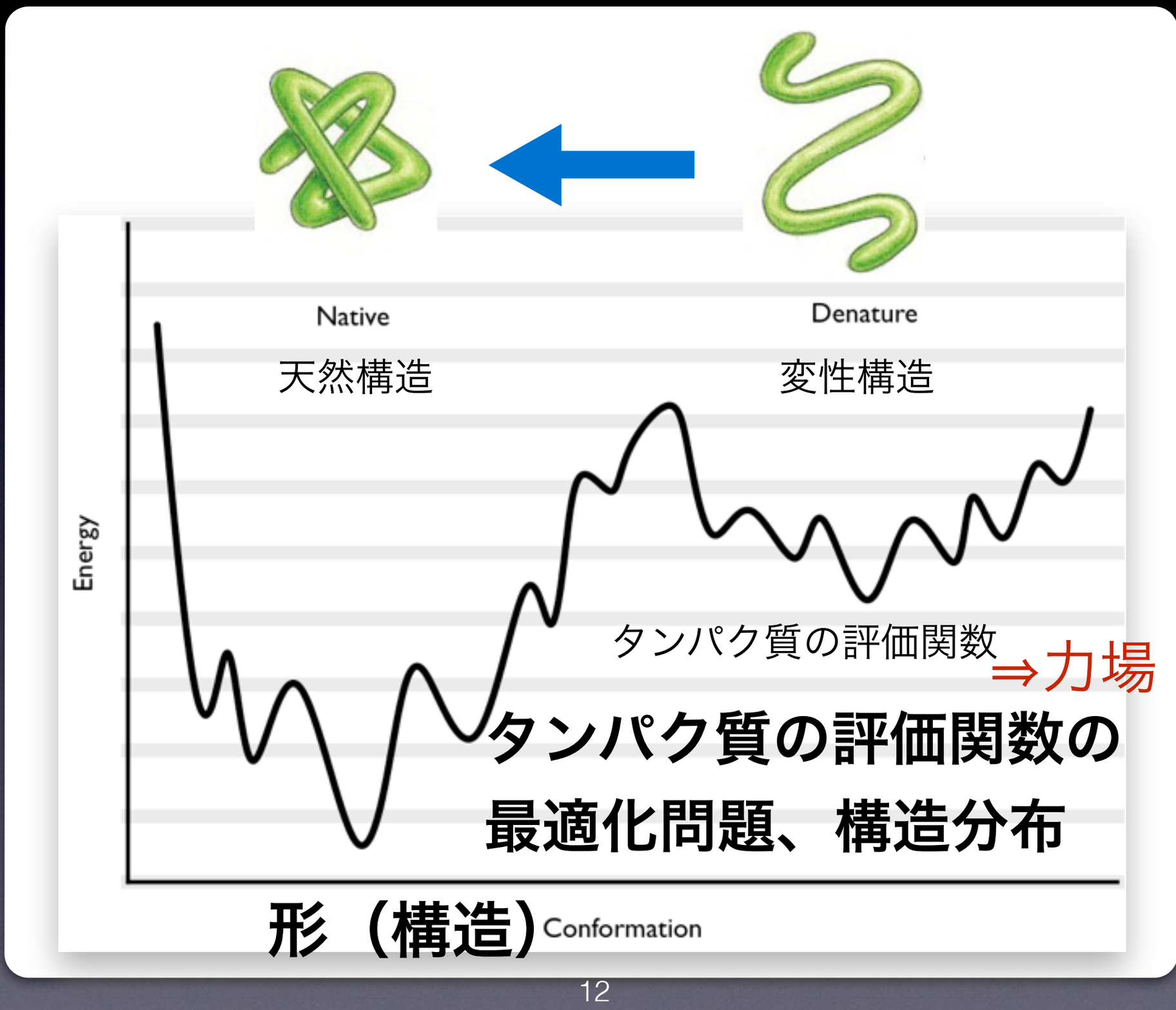
分子動力学法

- 力に沿った運動に従った（古典力学に従った、ある意味決定論的）、時系列データを得る事ができる方法（**ダイナミクスが追える**）。

- アンサンブル発生装置
  決められた条件（アンサンブル）下のデータを得る事ができる
  ⇒　**サンプリングデータとして利用できる。**

タンパク質の構造予測など

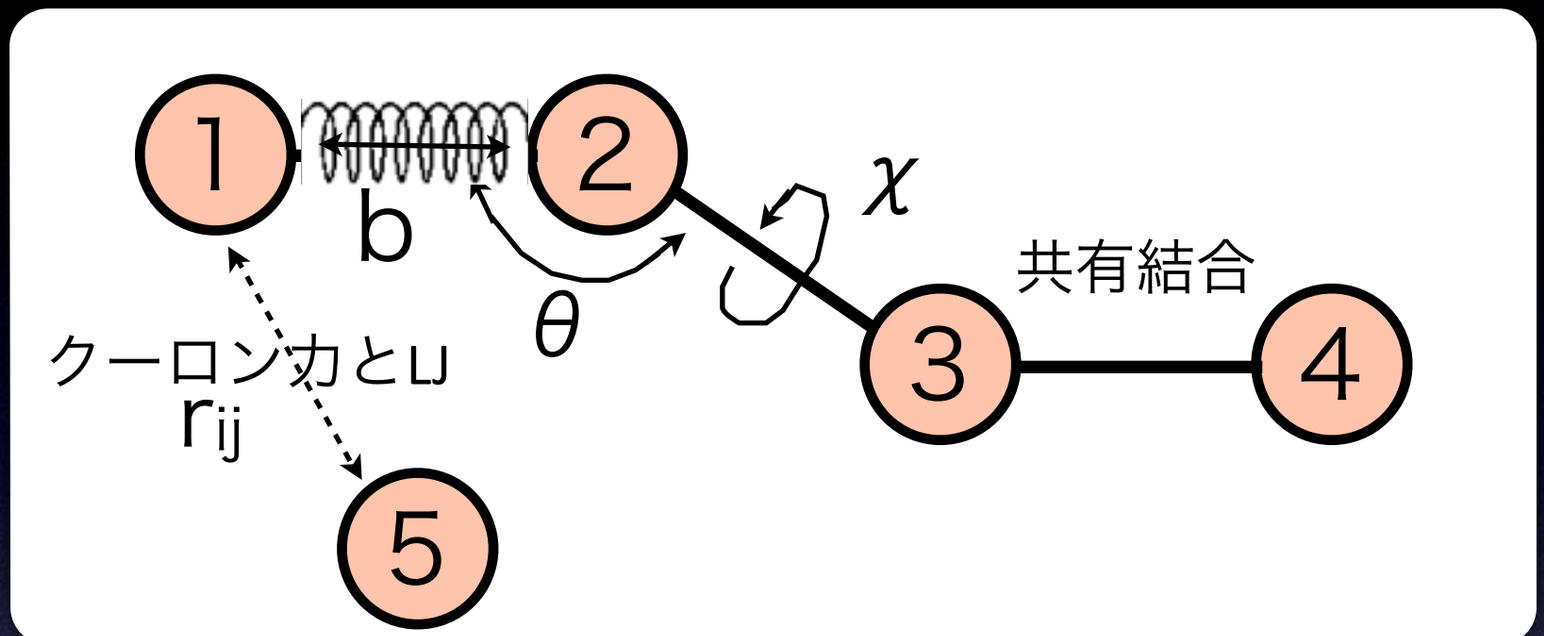
ここでは、構造揺らぎ・構造配置（ドッキング）・構造変化も含めた広い意味で



タンパク質の評価関数（力場）

バネだと考える

ビーズ１個は原子１個



$$H(q,r) = K + V$$

$$V = V_{internal} + V_{nonbond}$$

力学

$$V_{internal} = \sum_{\text{bonds}} K_b(b - b_0)^2 + \sum_{\text{angles}} K_\theta(\theta - \theta_0)^2 + \sum_{\text{dihedrals}} K_\chi[1 + \cos(n\chi - \sigma)]$$

$$V_{nonbond} = \sum_{\text{non-bonded atom pairs}} (4\epsilon_{ij}[(\frac{R_{min,ij}}{r_{ij}})^{12} - (\frac{R_{min,ij}}{r_{ij}})^{6}] + \frac{q_i q_j}{\epsilon_D r_{ij}})$$

電磁気学

L-J項 **クーロン項**

さまざまな力場と力場パラメータ

力場(FF)	version	得意分野
CHARMM	C19, C22, C27, C27-CMAP, C36　など	Lipid-protein interaction
AMBER	Parm94, Parm99, Parm99SB, 03 など	Protein, Glycan
GROMACS	43a1, 53a6, 53a7 など	Protein, Lipid
その他	OPLS FF, encad　など	

- 評価関数が微妙に違う
- パラメータの決め方が違う
- 得意不得意がある
- 最近はどれも結構良い

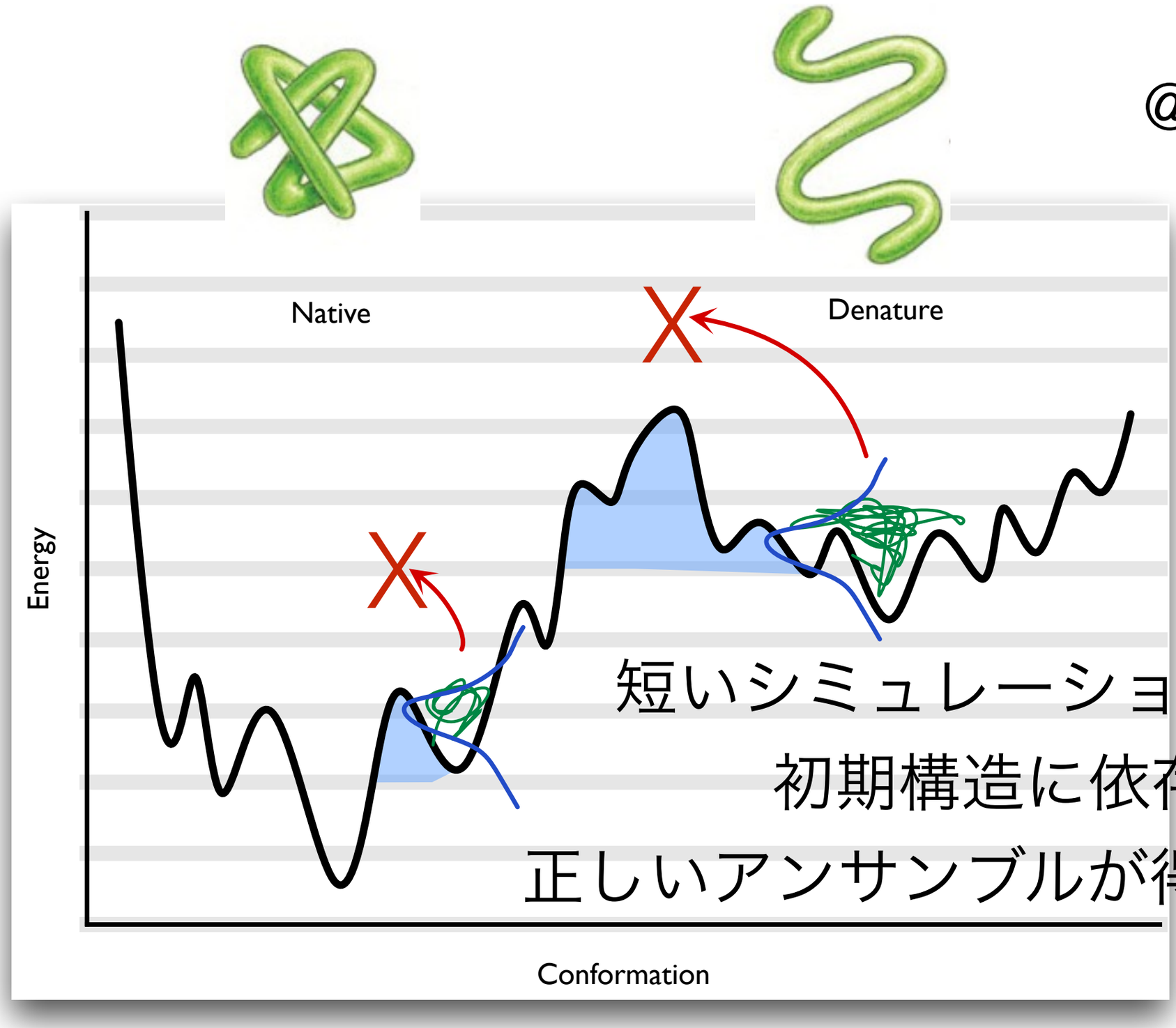
よく使われる MD パッケージ

MD Packages	Hybrid	Speed			
CHARMM [1]	×	Slow	多機能		
AMBER [2]	×	Medium/Fast	GPGPUで速い		
NAMD [3]	○	Fast	Charmの高速版 (GPGPU版もある)		
GROMACS [4]	○	Fast	1CPU性能高い/高速, GPL (GPGPU版も)		
DESMOND [5]	○	Fast	高速		
TINKER [6]	×	Slow	コードが読み易い		
MARBLE [7]	○	Fast	高速、京コンピュータに最適化		
GENESIS [8]	○	Fast	高速、京コンピュータなどに最適化		

[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
[8] J. Jung et al., (2015)

- **それぞれのパッケージに特徴・特色**
- **機能も違う**
- **計算スピードも違う**
- **力場も違う（近年、対応できる様になってきた）**

分子動力学法



@300K

短いシミュレーションだと

初期構造に依存

正しいアンサンブルが得られない

ハミルトニアンレプリカ交換法

Adaptive Tempering法

Wan-Landau法

アンブレラサンプリング

Meta-dynamics法

拡張アンサンブル法

レプリカ交換法

マルチカノニカル法

Tsallis統計を用いた方法

レプリカ交換分子動力学法

Adaptive Biasing法

レプリカ交換法はMCMC法

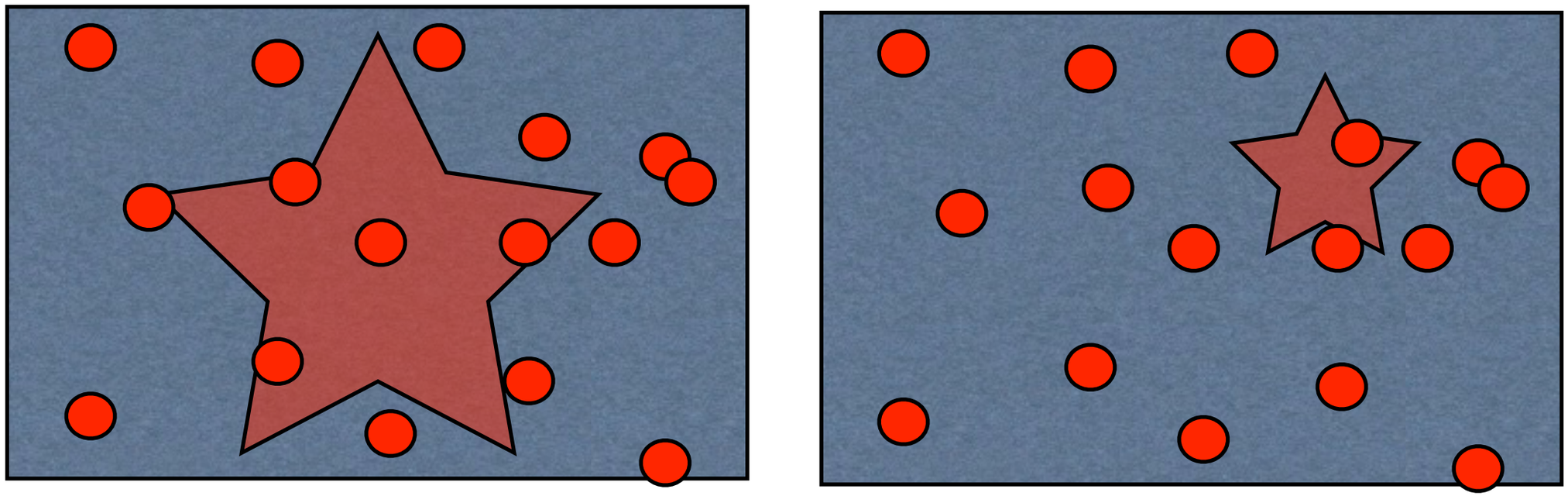
普通のモンテカルロ法：
静的モンテカルロ法

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

・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

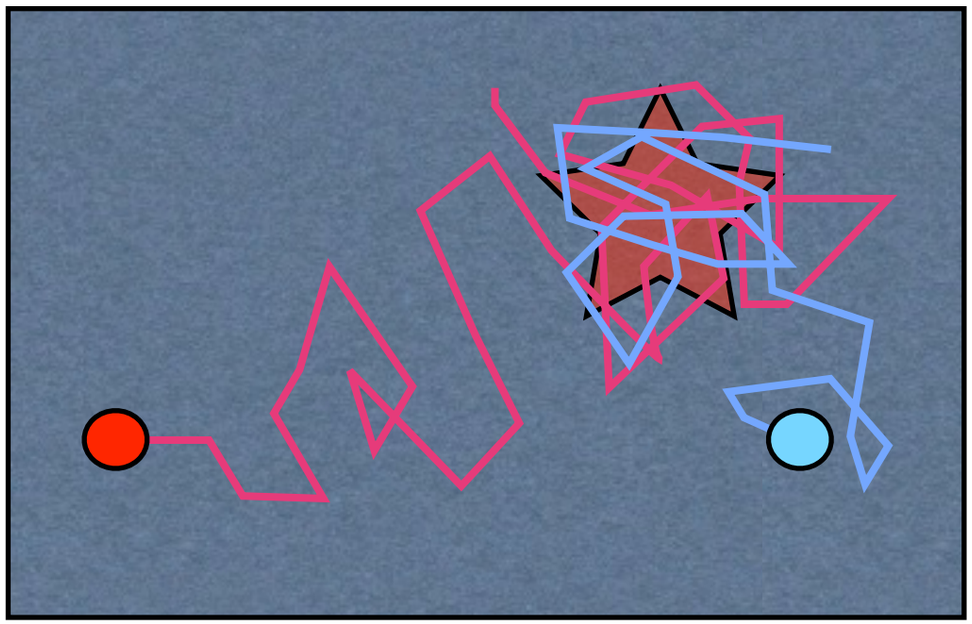
普通のモンテカルロ法

システムの次元が大きくなると、相対的にターゲットが小さくなり、ダーツがヒットしにくくなる



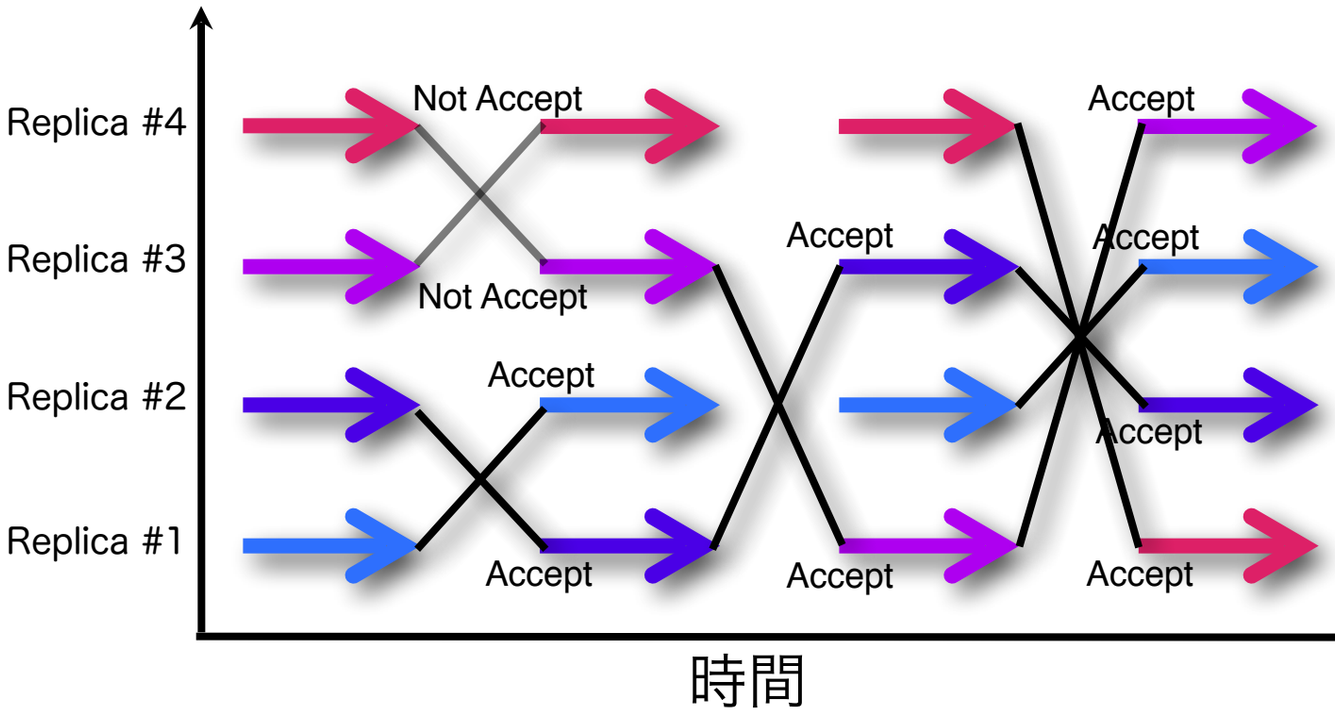
タンパク質の構造予測：多次元系での構造探索

→マルコフ連鎖モンテカルロ法



- どこからか、出発して、だんだん確率の大きい方に移動し、確率が大きくなったら、その近傍を探索する方法がよい
- 更に、並列で探索(REMD)

レプリカ交換分子動力学 (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)

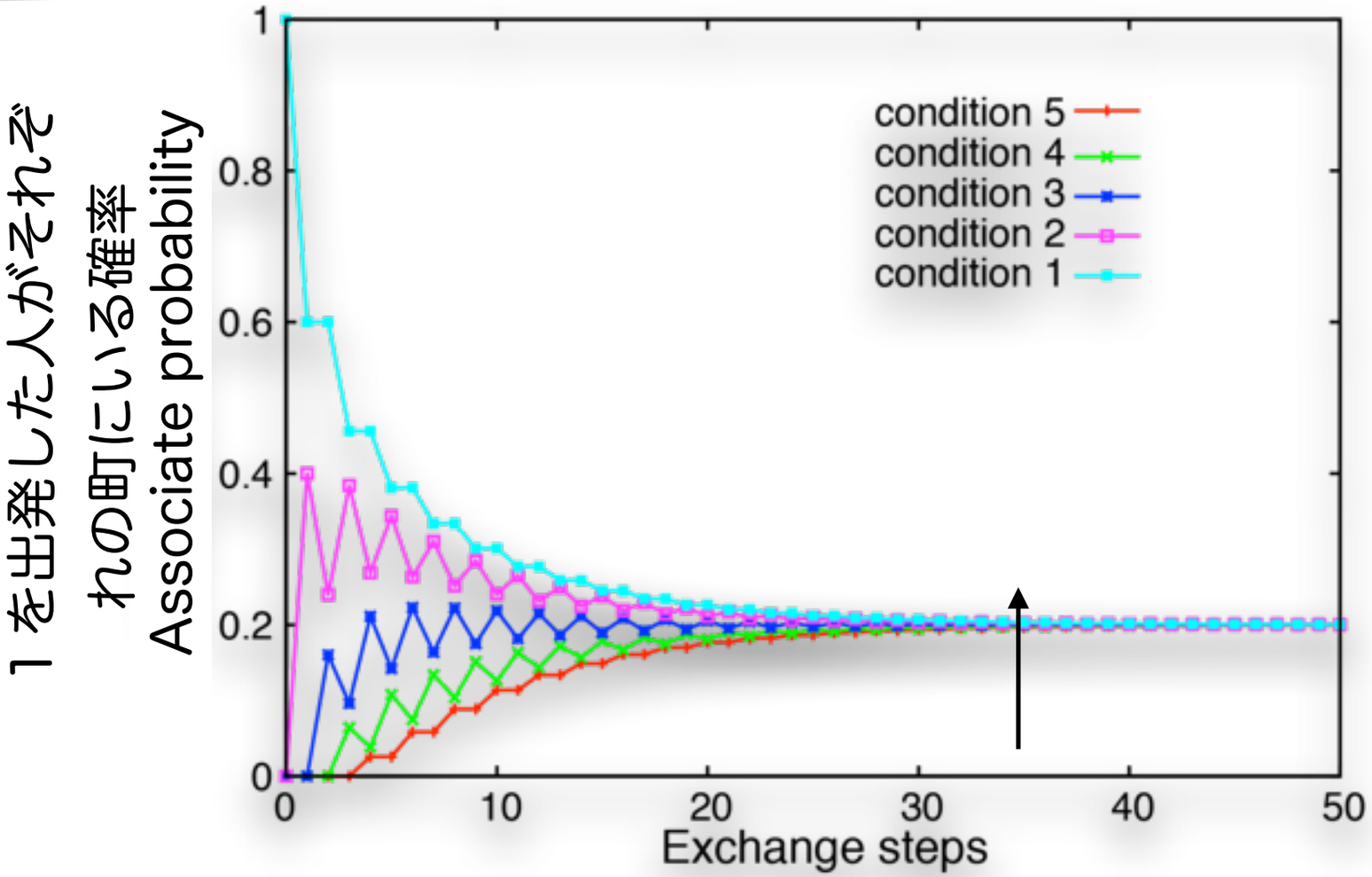
MCMCの条件

- 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。

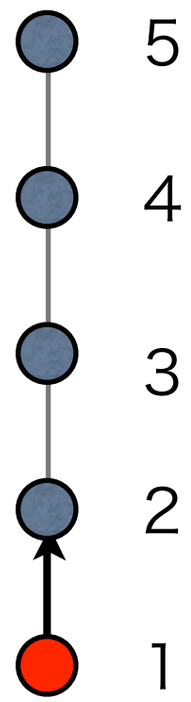
定常マルコフ過程

保守的な５つの町で商売をしている塩商売人の話。
一本の道でつながっており、関所と門で交通制限されている５つの町の往来を考える。
町には受理率40%の関所がある。町には南北２つの門（関所）があり、１日毎に開門される門が替わる。１番町は北門、５番町は南門のみで１日毎に開け閉めがある。関所破りは厳罰。

許されている道路
１日目：1-2, 3-4
２日目：2-3, 4-5



1/N



35日くらいで全ての町を満遍なく回れる。

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

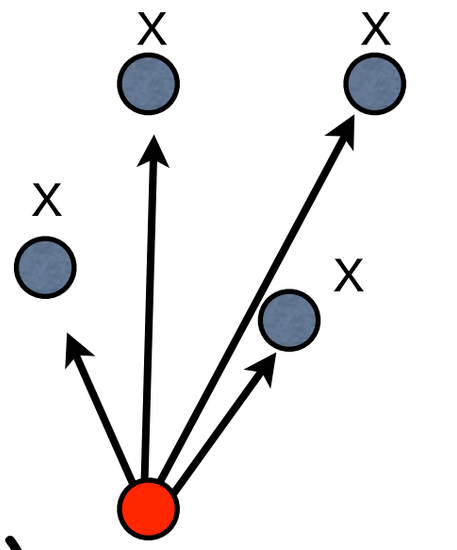
$$R(x|y)f(y) = R(y|x)f(x)$$

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。

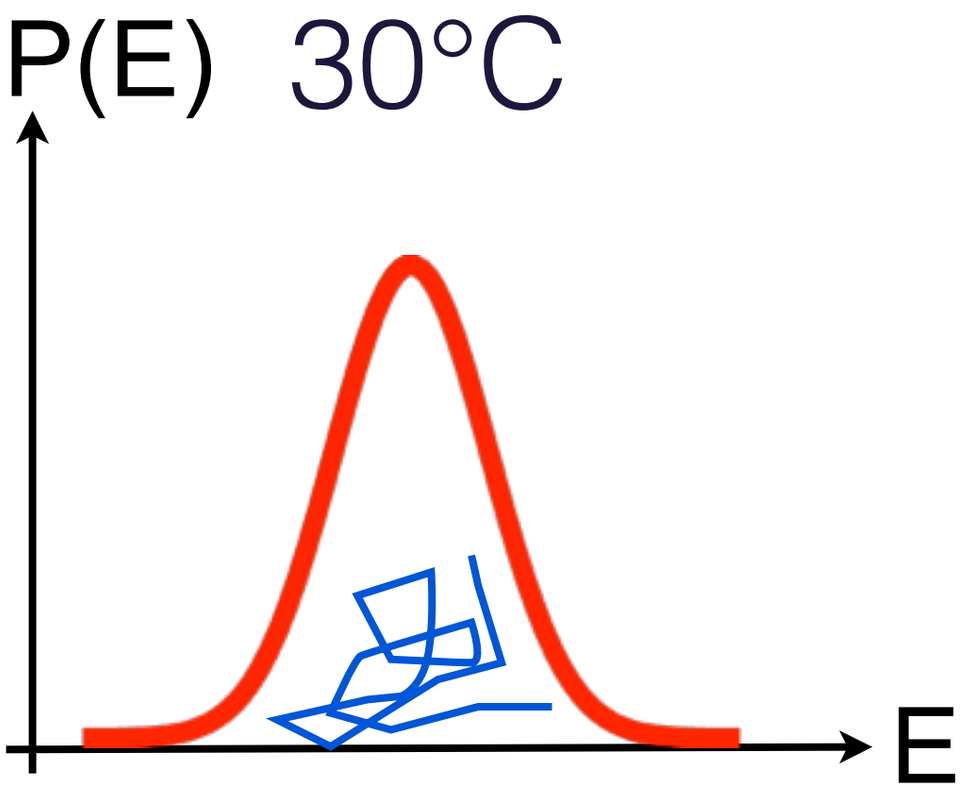
$$\langle K(p) \rangle_T = \left\langle \sum_{k=1}^{N} \frac{p_k^2}{2m_k} \right\rangle_T = \tfrac{3}{2} N k_B T .$$

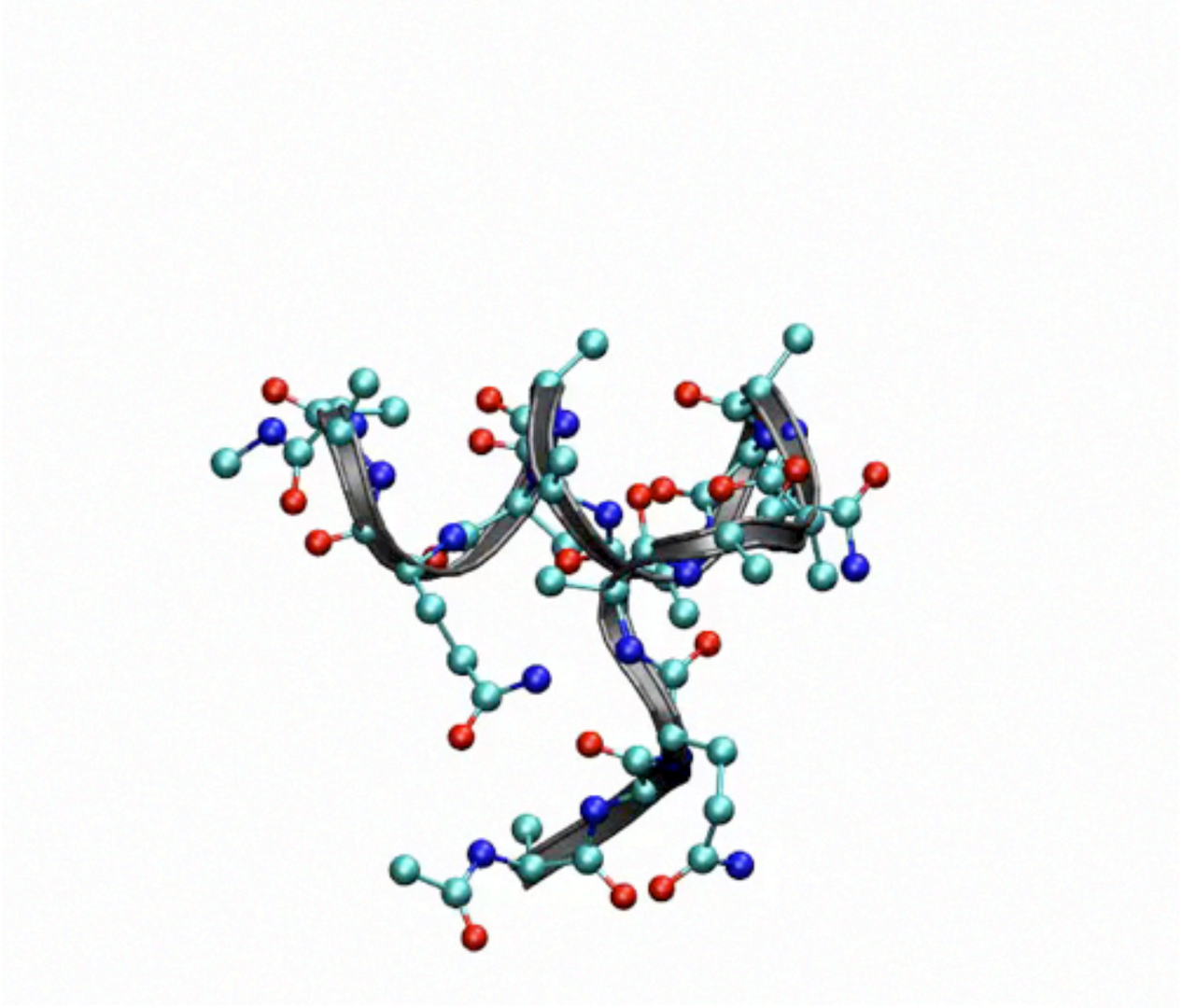
$$\begin{cases} p^{[i]'} \equiv \sqrt{\dfrac{T_n}{T_m}}\, p^{[i]}, \\ p^{[j]'} \equiv \sqrt{\dfrac{T_m}{T_n}}\, p^{[j]}, \end{cases}$$

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）

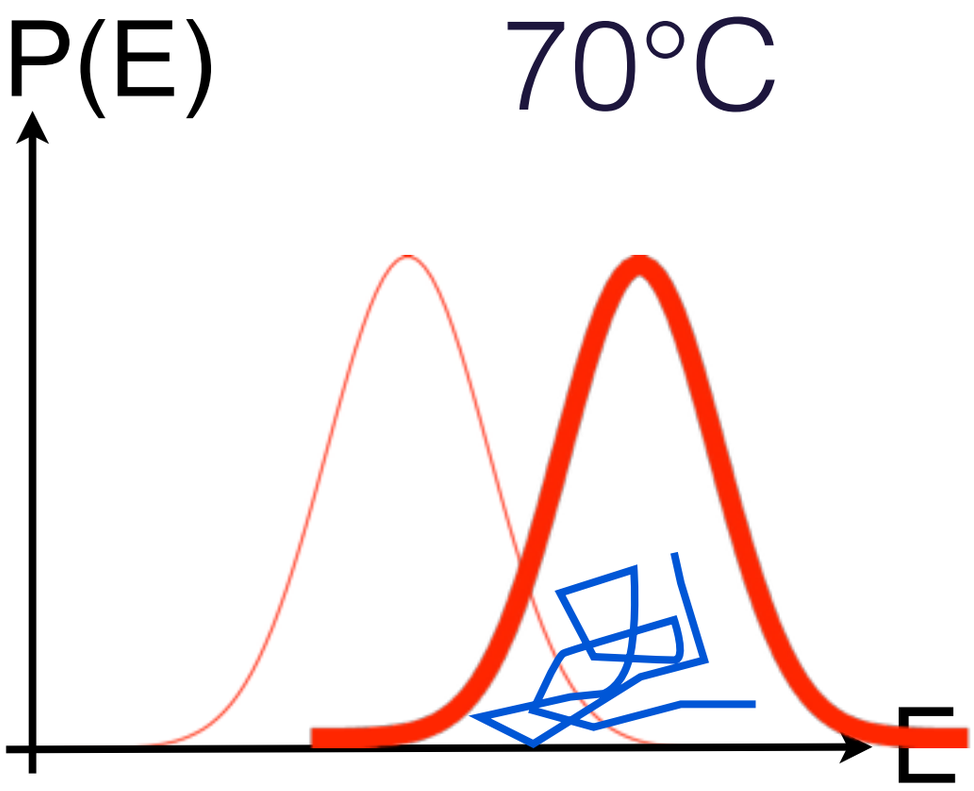


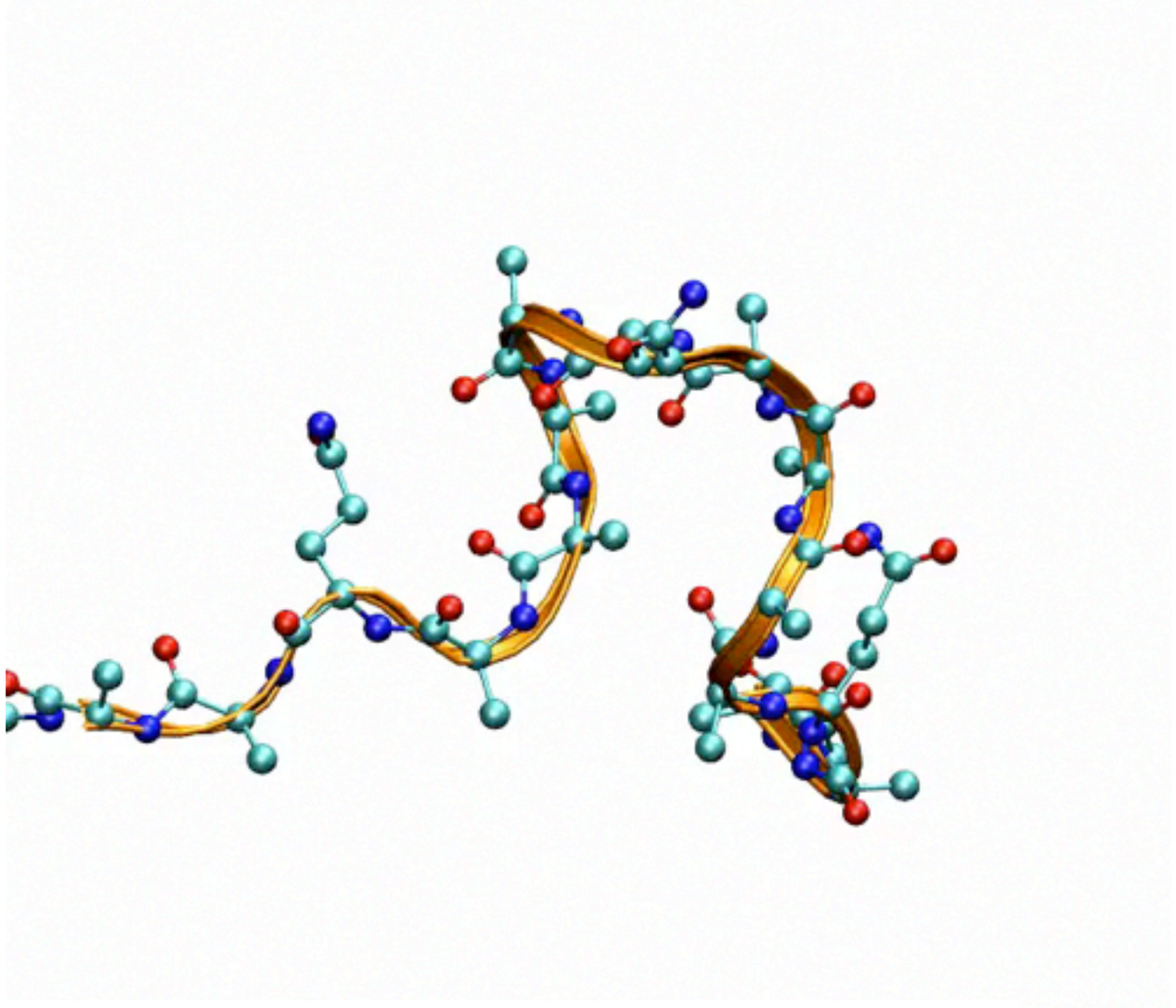


エネルギー空間、局所最小値に捕まる

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

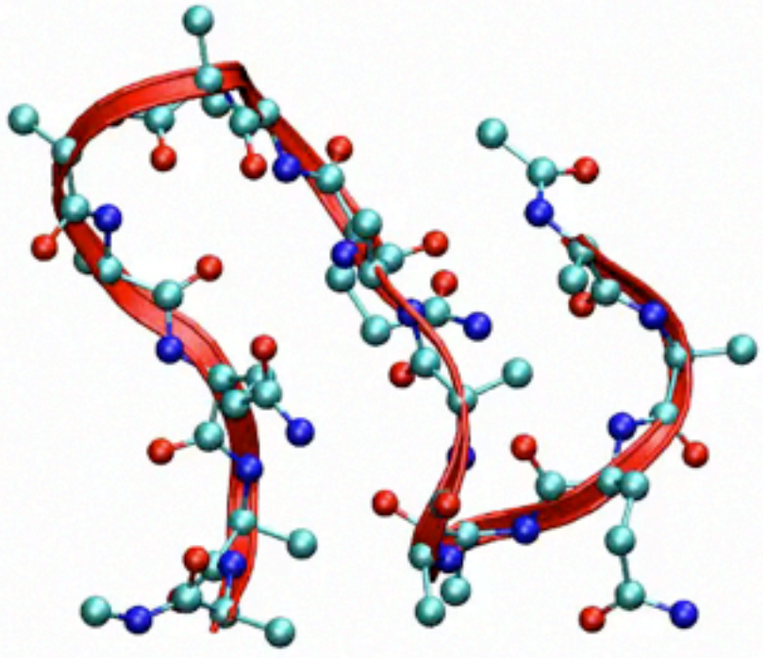


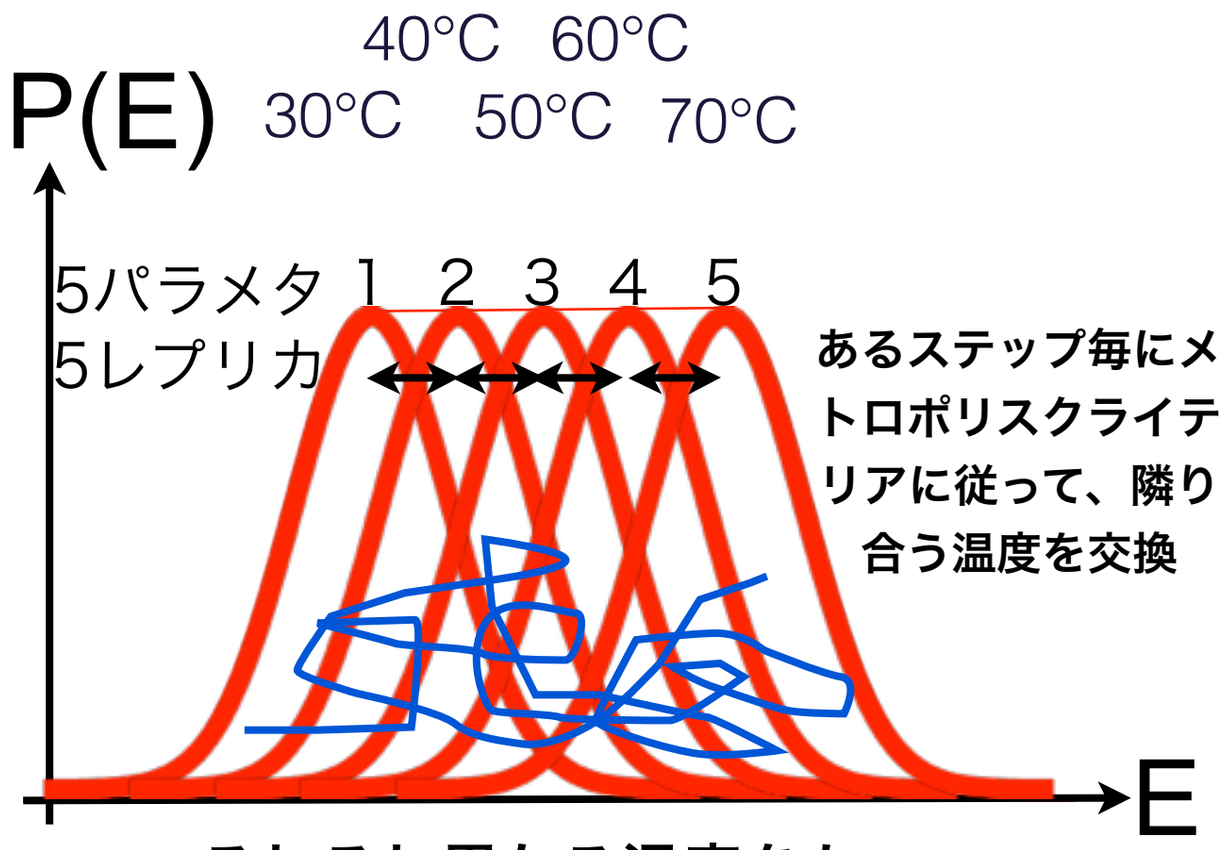


エネルギー空間、局所最小値に捕まらないが

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

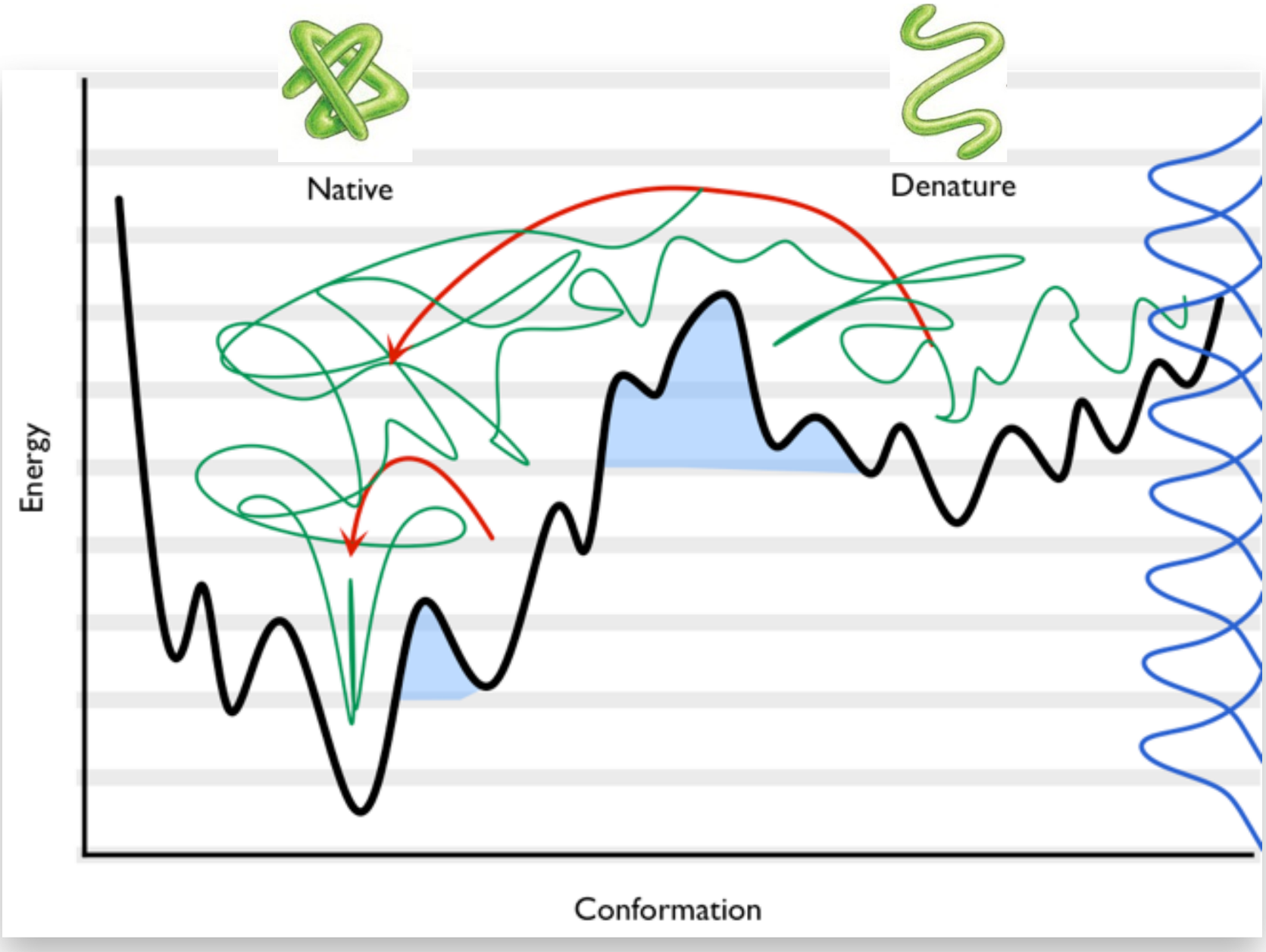
複数のレプリカを用意しMD計算

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

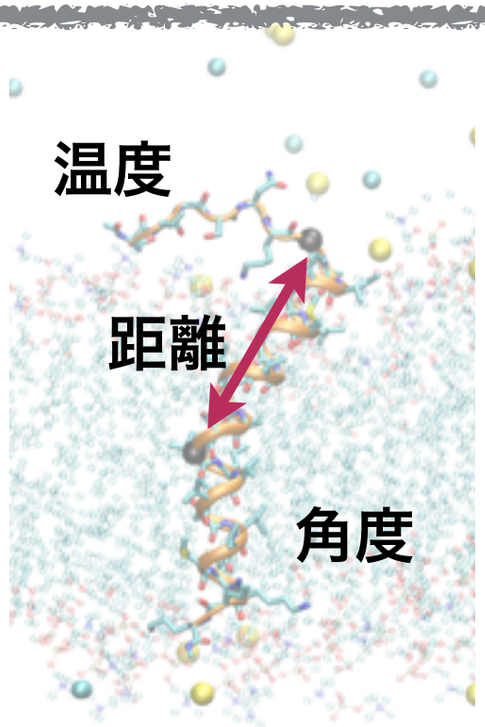
バリアを乗越える事ができる

温度の異なるMDを行い、ピコ秒毎に温度を交換

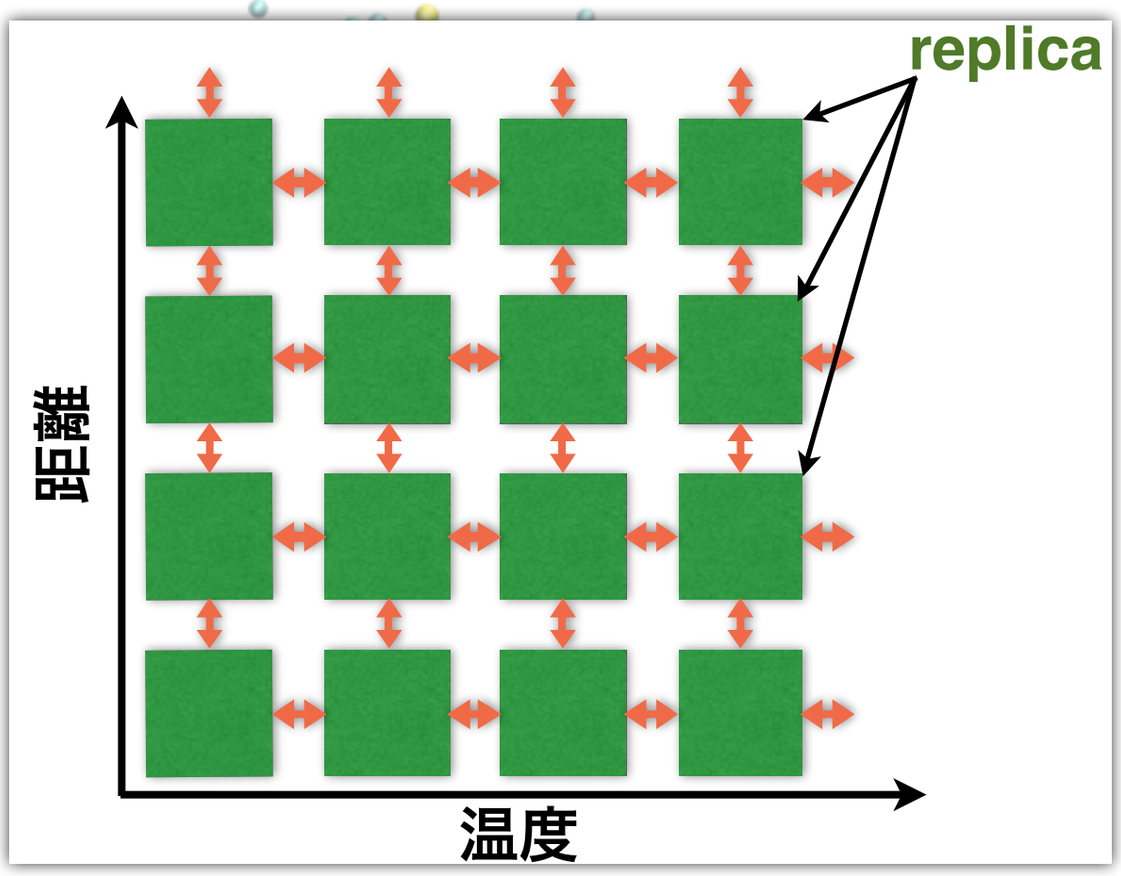


@100℃
@90℃
@80℃
@70℃
@60℃
@50℃
@40℃
@30℃

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)

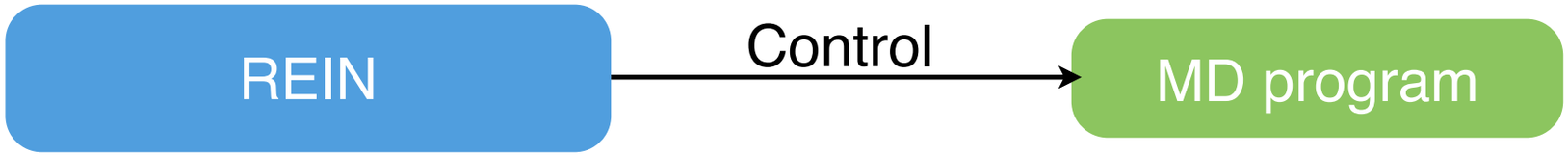
京コンピュータの沢山のノードを利用した計算が実現できる。

2. REIN-Kプログラムの原理

REINプログラム

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

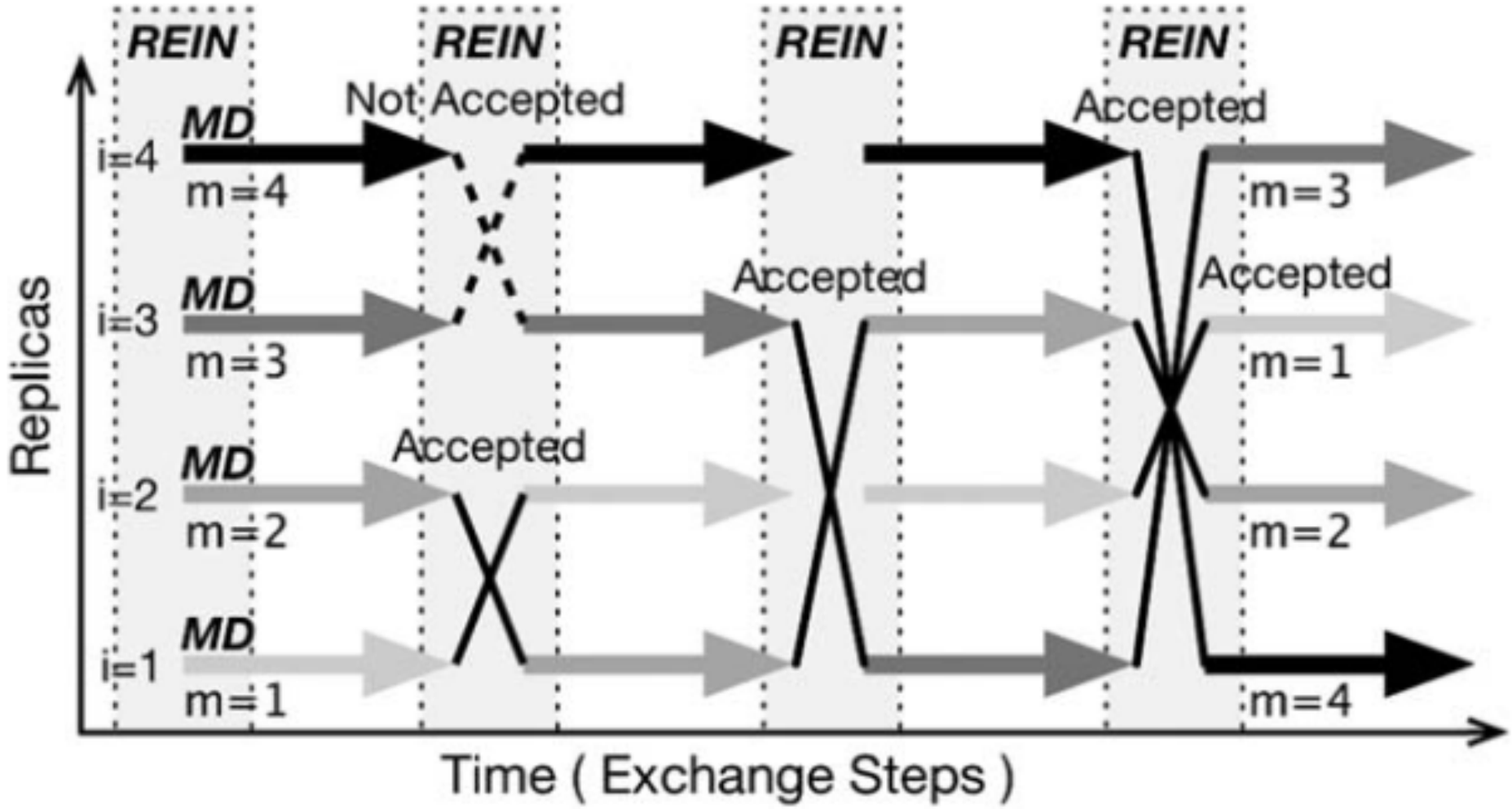
REINはバイナリの外部プログラムを制御します



REIN：親プログラム

バイナリ外部プログラム(MD)：子プログラム

- 子プログラムの実行
- 子プログラムの終了判定
- 子プログラムのoutputの読込み
  - エネルギー等の情報取得
  - レプリカ交換実行
  - 子プログラムの次のinputを作成



N. Miyashita et al., *IJQC*, **115**(5), 325–332. (2015)

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	MPI-2 version	MPI-1 version
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

レプリカ交換の２つの仕組みを用意

MPI-1 version

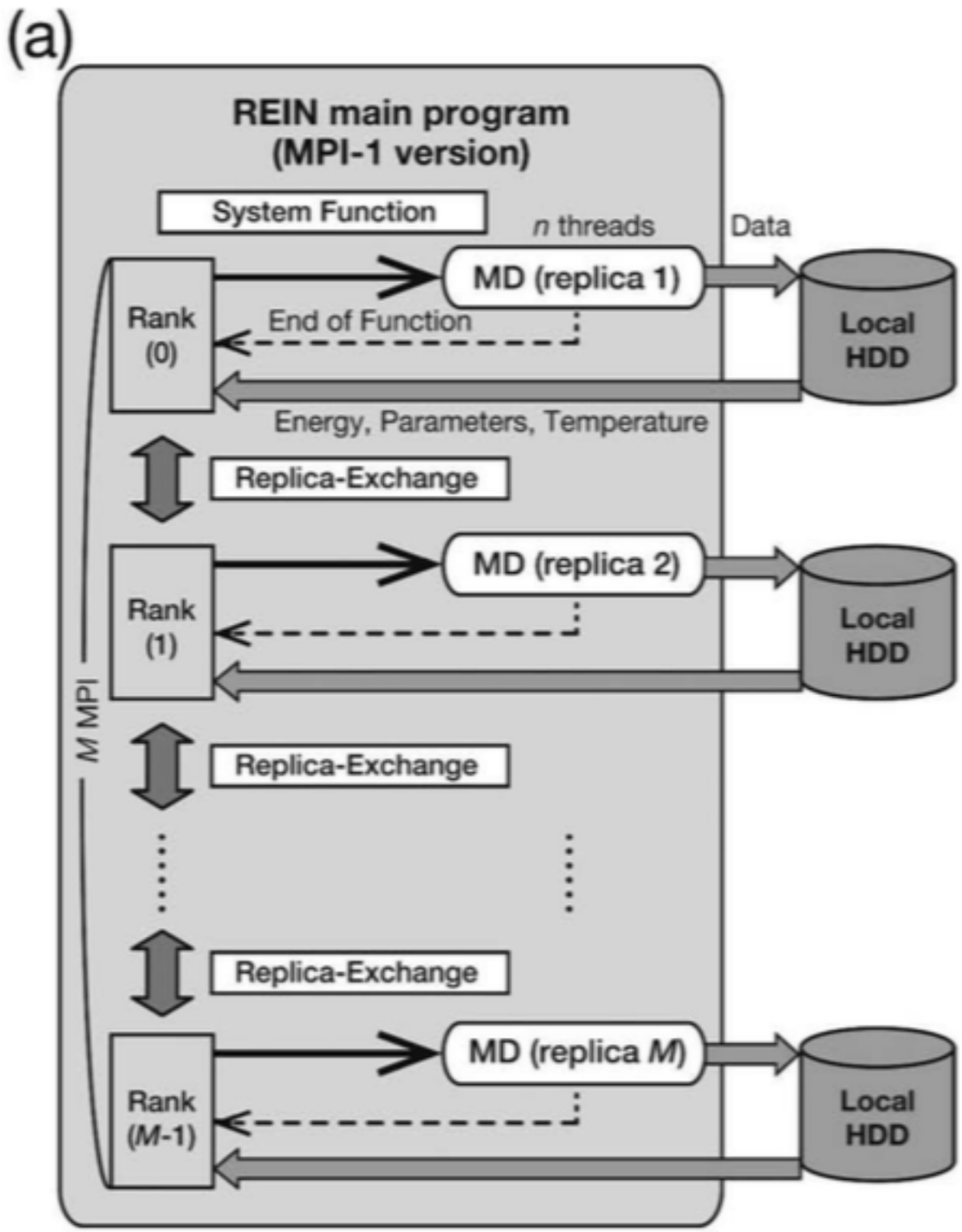
一般的なPCクラスタ、MDは1ノード以内

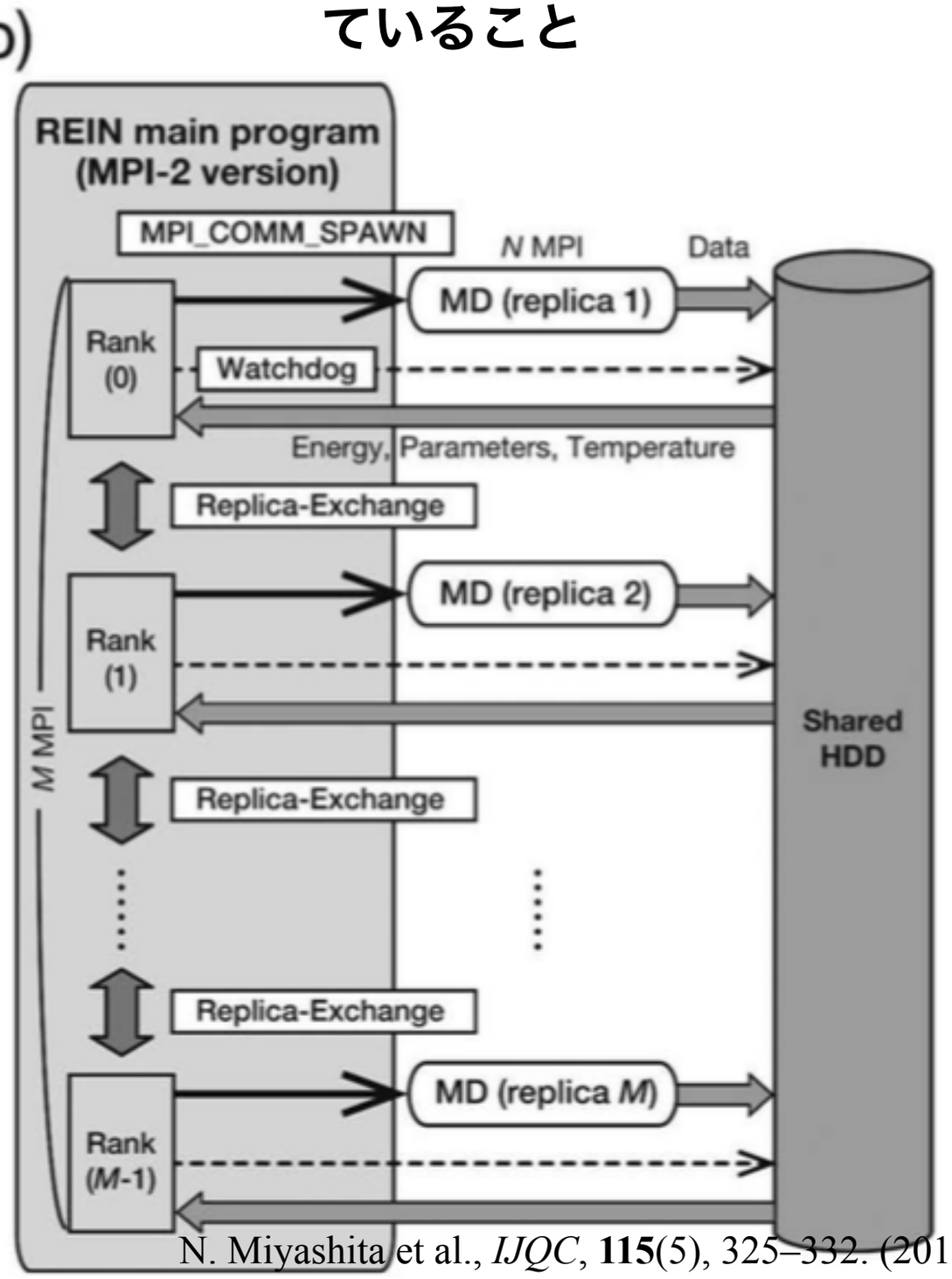
MDはMPI並列でない場合

MPI-2 version

高度なPCクラスタ、MDは1ノード以上も可

MDはMPI並列化もしくはHybrid並列化されていること





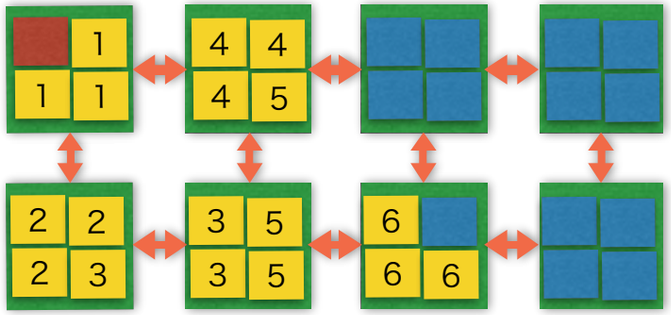
N. Miyashita et al., *IJQC*, **115**(5), 325–332. (2015)

MPI-2 version
MPI_COMM_SPAWN

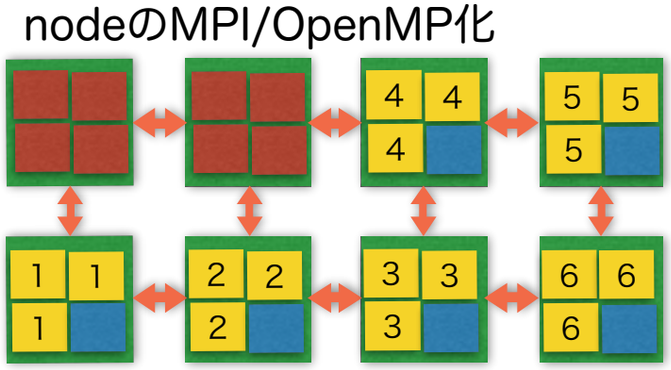
- 動的プロセス生成
  - MPIのプロセスからMPIを実行する
  - MPI-2に対応していればどの計算機でも動く
  - スケジューラが制限をかけている場合がある
  - Master processが必要
  - Child Jobはnodeをまたぐ事ができる

例：4 core x 8 nodeのシステム

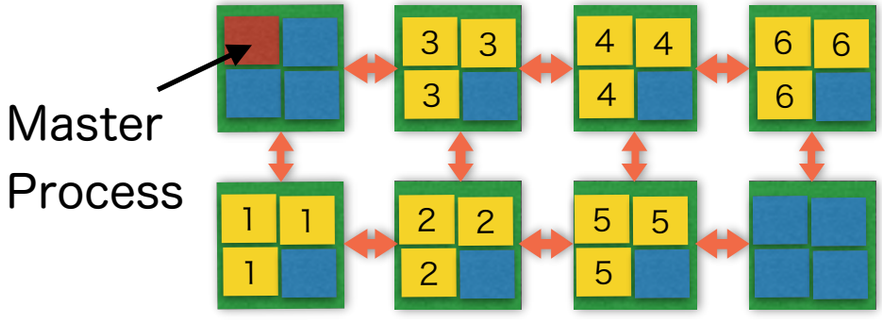
制限の無いPCクラスタ



京コンピュータ, FX10でMaster nodeのMPI/OpenMP化



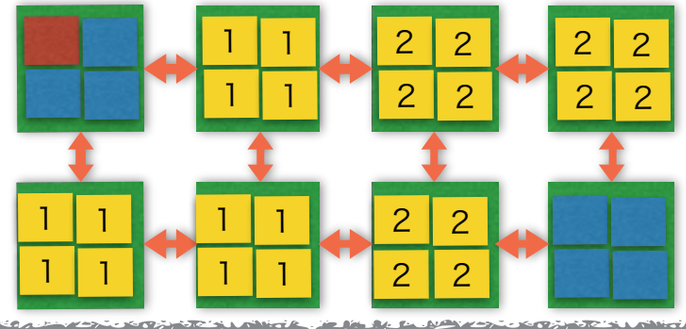
Master process: 1 core
Child Job: 3 core x 6 Job生成
京コンピュータ, FX10



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

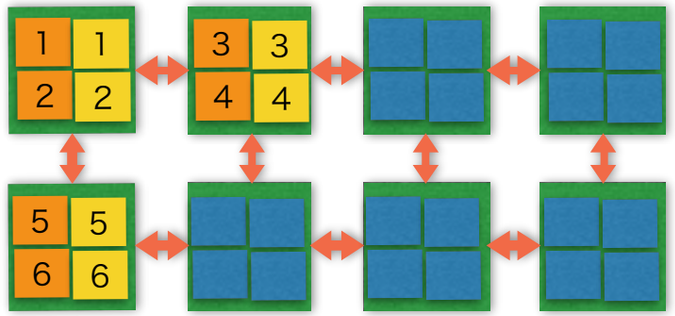


MPI-1 version
system 関数を用いたレプリカの制御

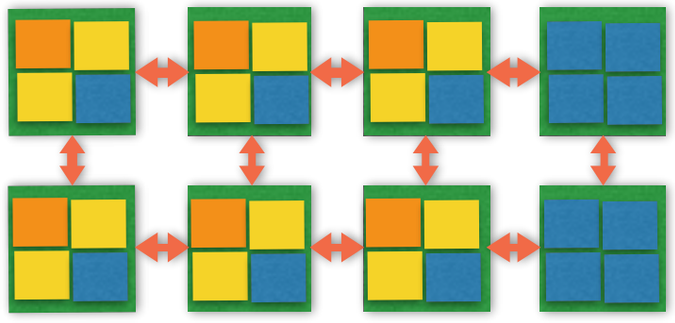
- MPIのプロセスからSystem関数を実行
- 大抵のPCクラスタで動く
- しばしばスケジューラが制限をかけている場合がある
- Master processは必要ないが、Child Jobの実行プロセスの１coreが制御にも使われる。
- 現状ではMPIプロセスを生成できない（nodeをまたぐ事ができない）

例：4 core x 8 nodeのシステム

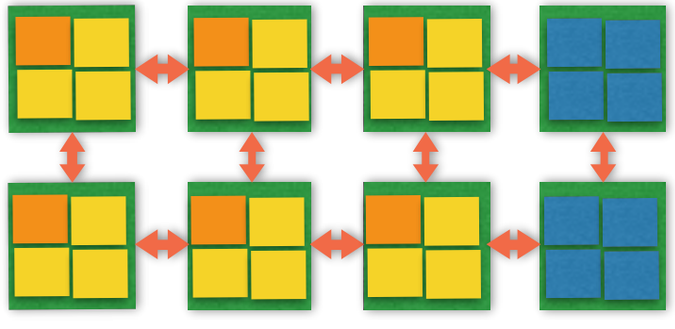
Child Jog: 2 core x 6 Job生成



Child Jog: 3 core x 6 Job生成

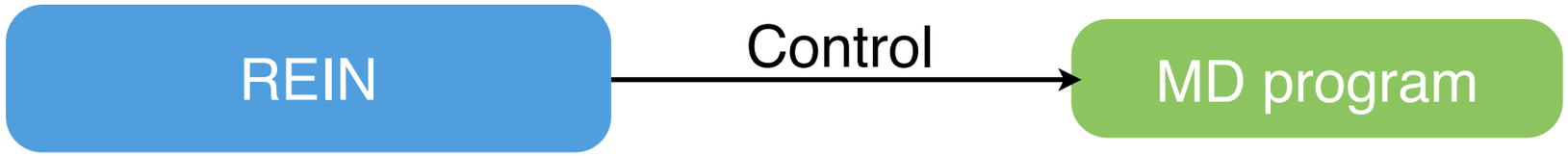


Child Jog: 4 core x 6 Job生成



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)/OpenMP

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	○	○	○	○	×	○	×

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

GPGPUを搭載するPCクラスターやPCクラスター

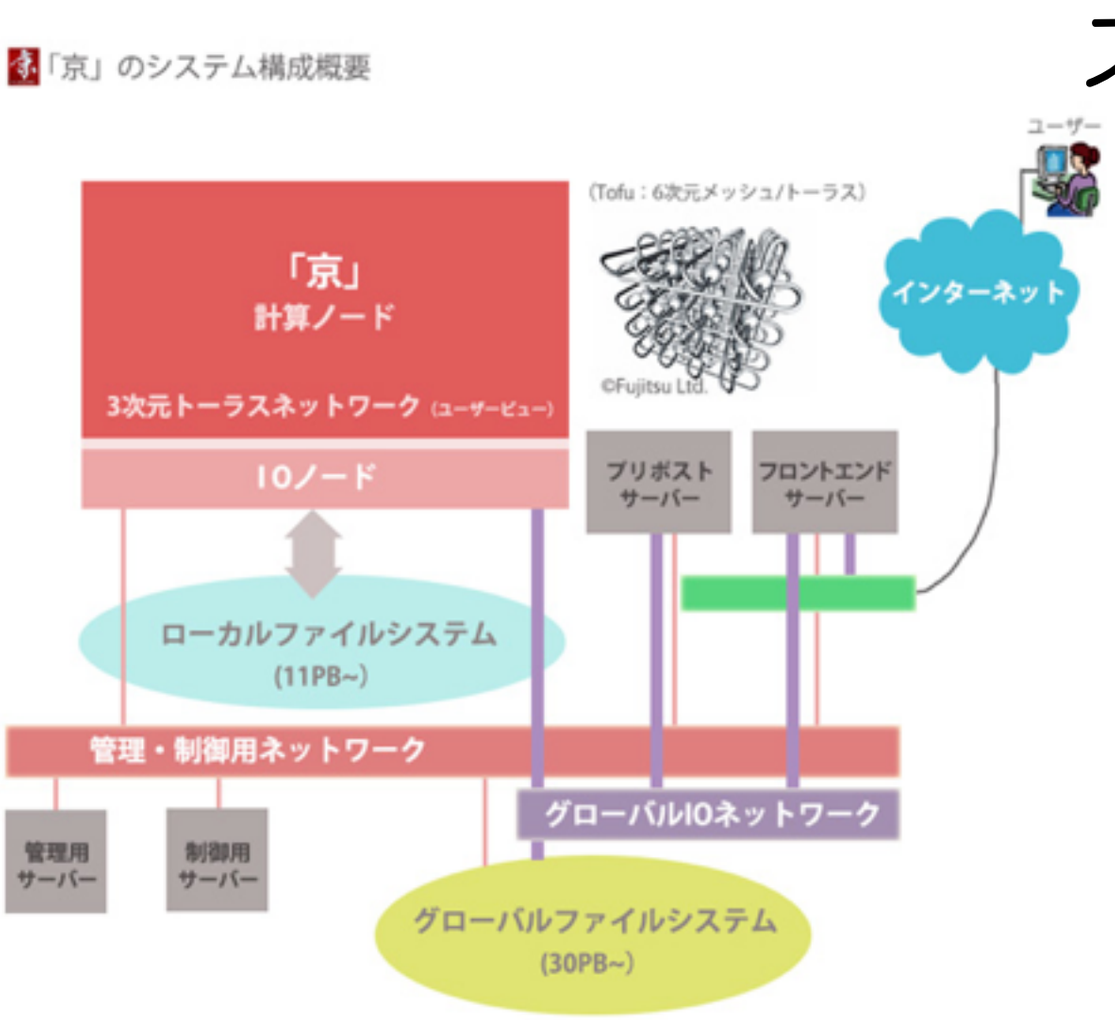
TSUBAME 2.5　　5.7 PFLOPS
@Tokyo Institute of Technology

東工大web pageより

1442 node x (12 cores + 3 GPGPU (K20X))

NAMD2.10 nighty version 4/24

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

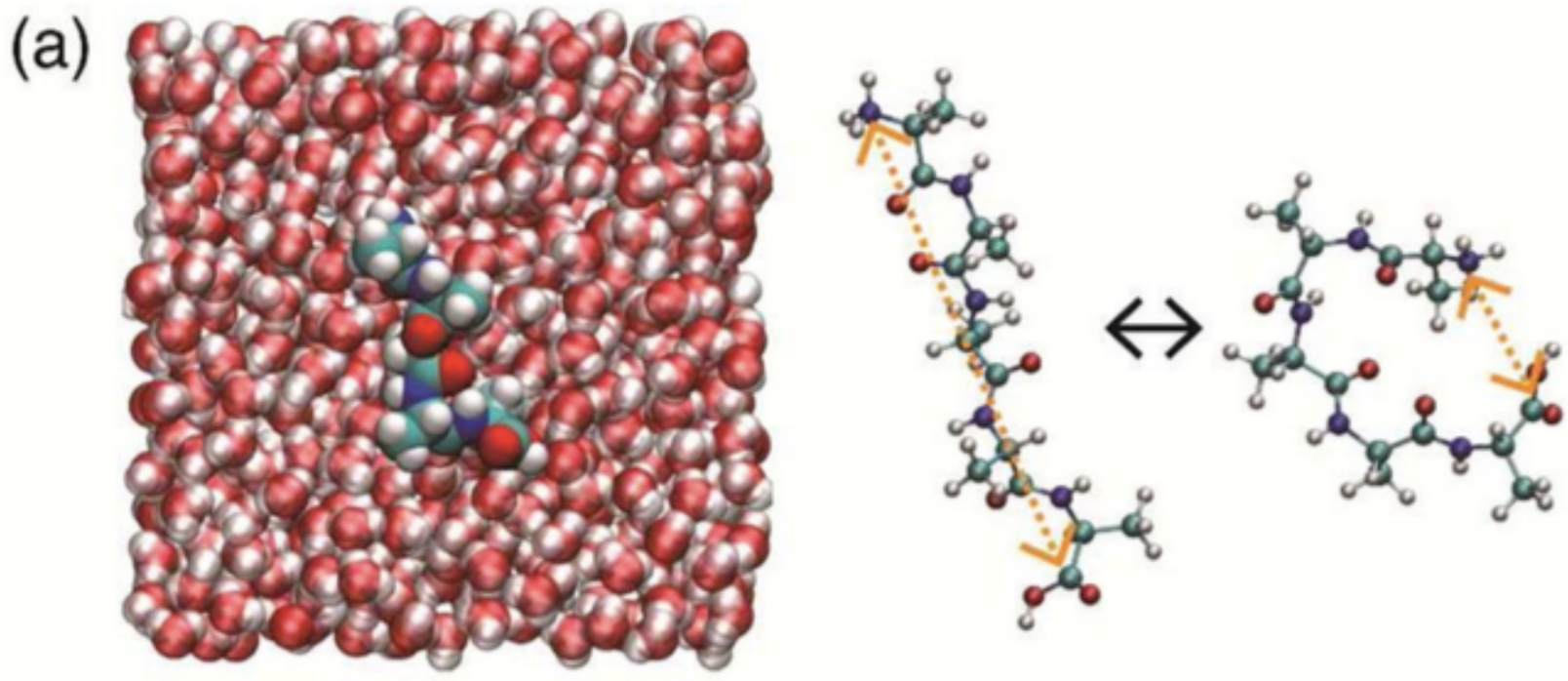
REIN プラットフォーム（計算科学）

モデル	REINはbinary形式のプログラムを京コンピュータやPCクラスタ(GPGPU対応)上で複数起動
機能	任意のパラメータを変えて並列に子プログラムを起動し、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 (対応子プログラム)	京コンピュータで動くnamd（とmarble）に対応 PCクラスタ(GPGPU対応可)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, TSUBAMEでも利用可
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。

REIN-Kを用いた
Penta-Alaninの構造予測の例



N. Miyashita et al., *IJQC*, **115**(5), 325–332. (2015)

レプリカパラメータ：温度とEnd-to-endの距離

	T-REMD	REUS	MREMD(2D)
温度	298-500K [32 replica]	300K	300-350K [4 replica]
距離	—	4-17 Å [12 replica]	4-17 Å [8 replica]
Harmonic restraining potential		1.0 kcal/mol/ Å^2	1.0 kcal/mol/ Å^2
レプリカ数	32	12	32
Exchange Trials	10,000 [Total 640ns]	10,000 [Total 240ns]	8,000 [Total 512ns]

2 ps for 1 exchange

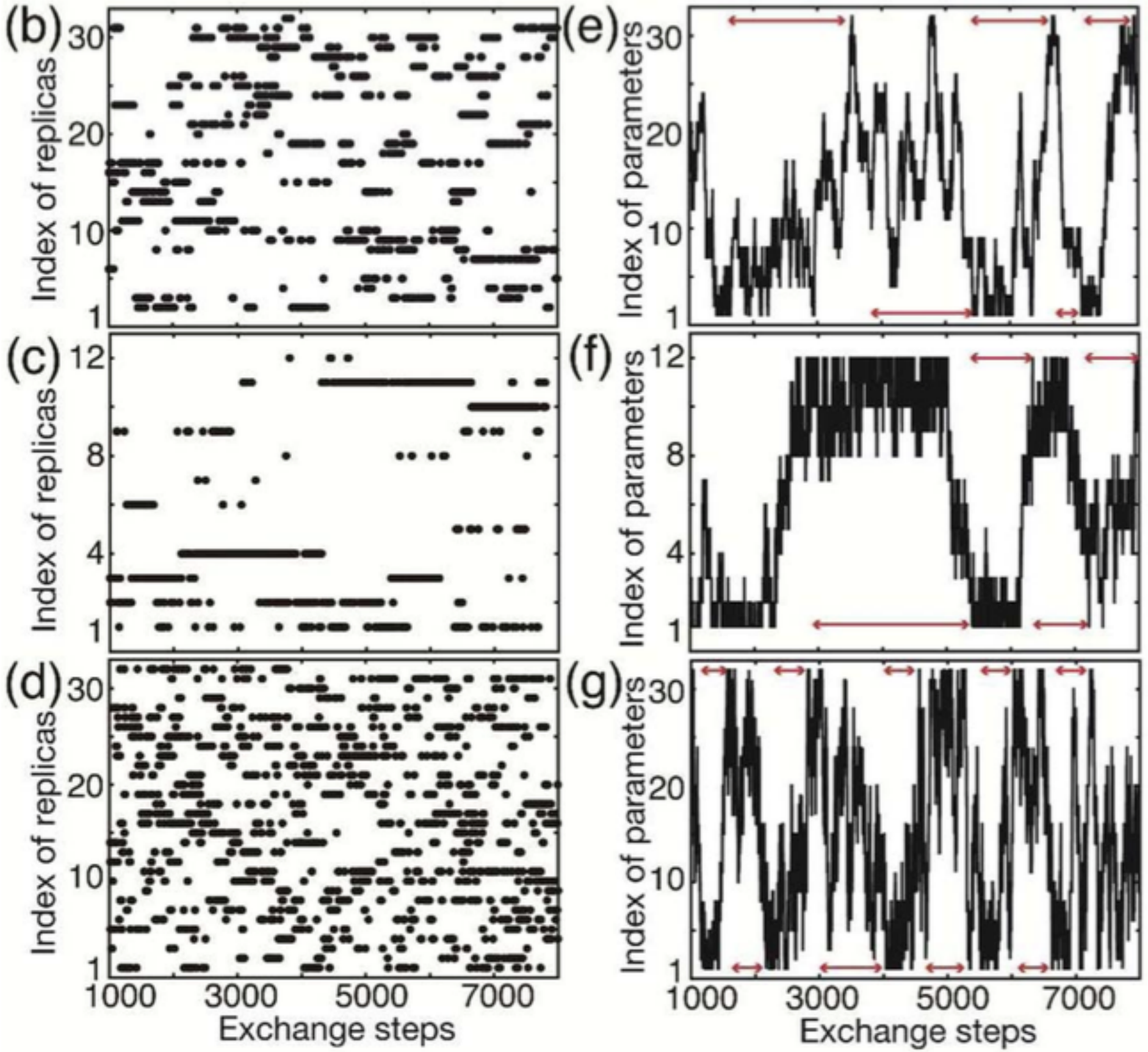
力場：charmm c36 CMAP

MD: NAMD2.9, MPI-2 version on K-computer

T-REMD

REUS

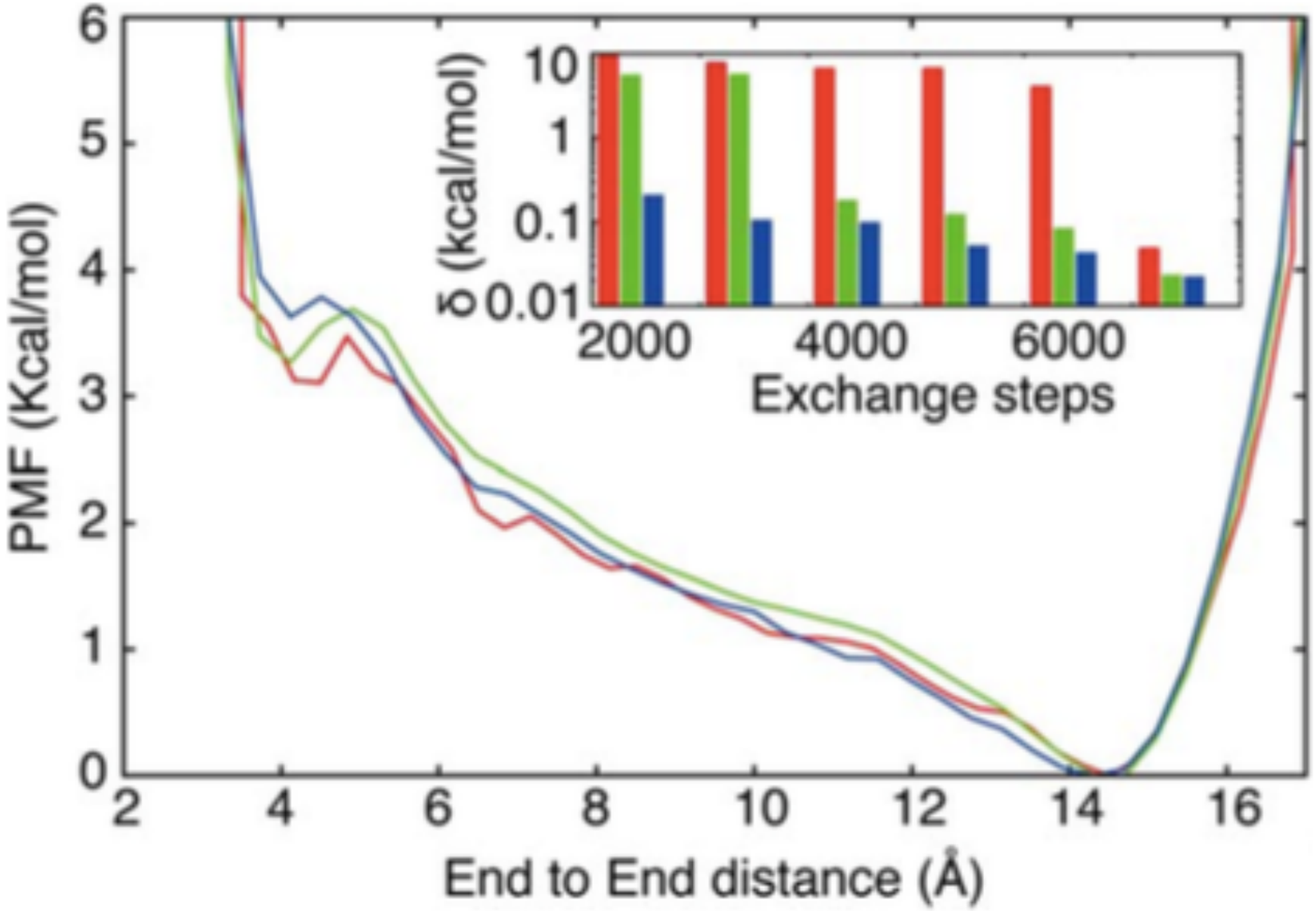
MREMD



N. Miyashita et al., *IJQC*, **115**(5), 325–332. (2015)

MREMDの収束性は非常に良い

WHAM @ 300K

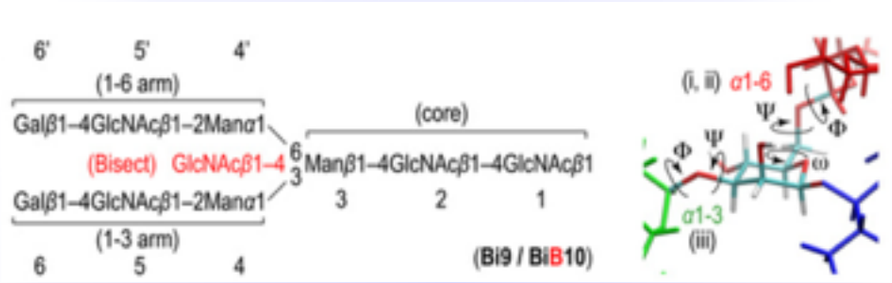


赤：T-REMD

緑：REUS

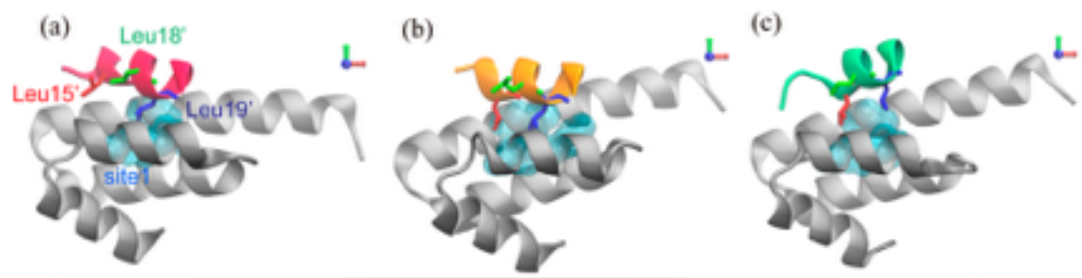
青：MREMD

REIN-Kを利用した生体分子構造予測シミュレーション



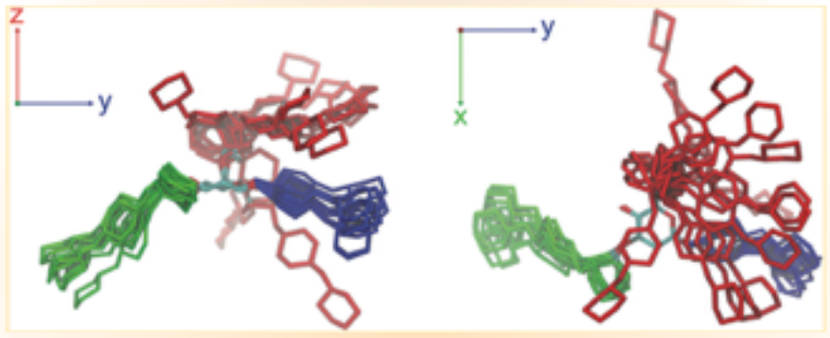
糖鎖の構造予測

Re, S., **NM** et al., 2011. *Biophys Journal*, 101(10), pp.L44–L46



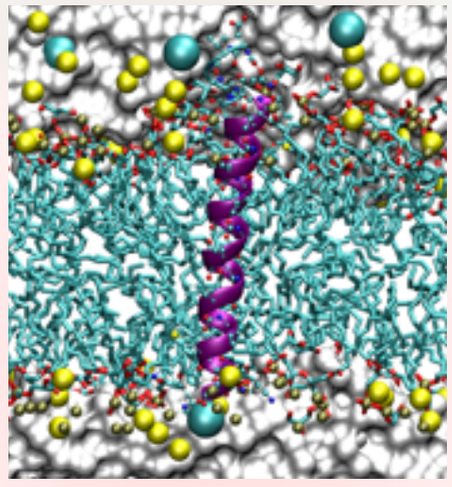
TOM20の機能解析

Komuro, Y., **NM** et al., 2013. *Journal Of Physical Chemistry B*, 117(10), pp.2864–2871.



糖鎖の構造予測２

Nishima, W., **NM** et al., 2012. *Journal Of Physical Chemistry B*, 116(29), pp.8504–8512.



膜タンパク質の構造予測

さまざまな生体分子の構造予測に利用できる

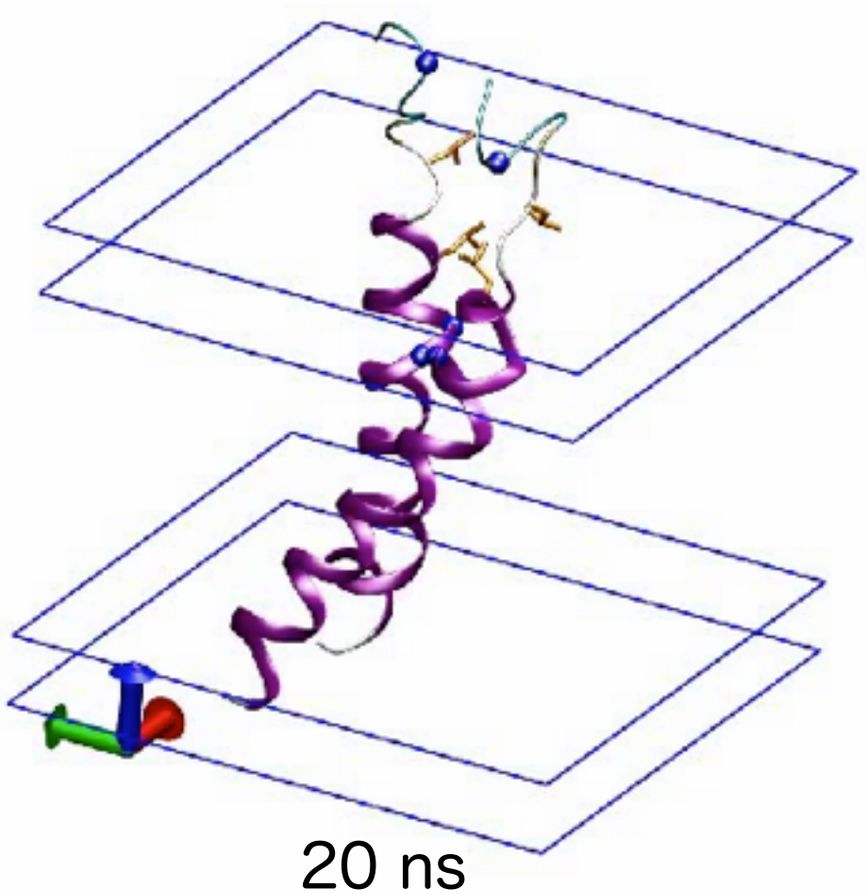
3. レプリカ交換分子動力学法を用いた実際の研究例

膜タンパク質の二量体構造予測

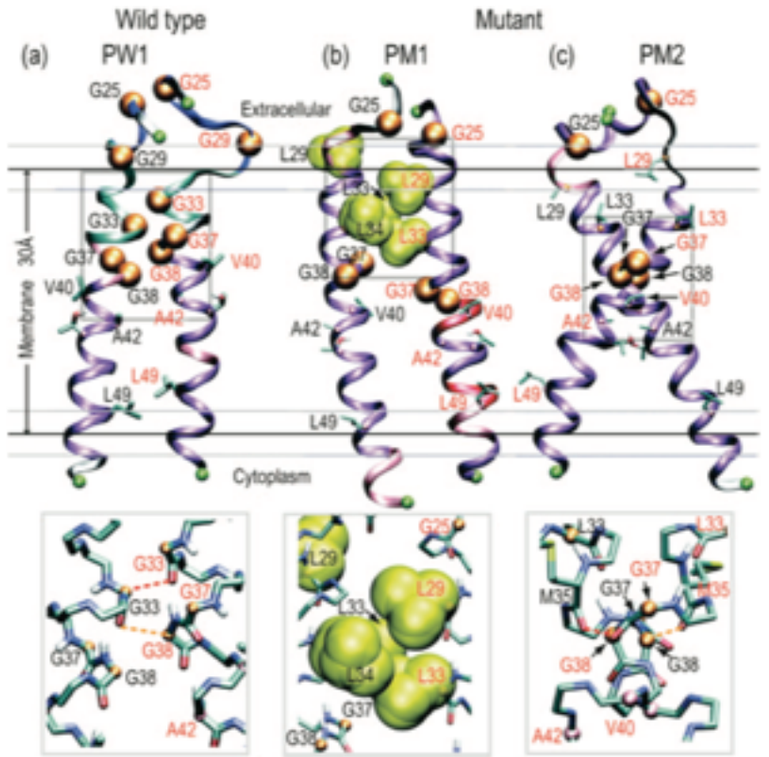
アミロイド前駆体タンパク質の二量体構造予測
インターフェースの変化がAβペプチド生成に影響

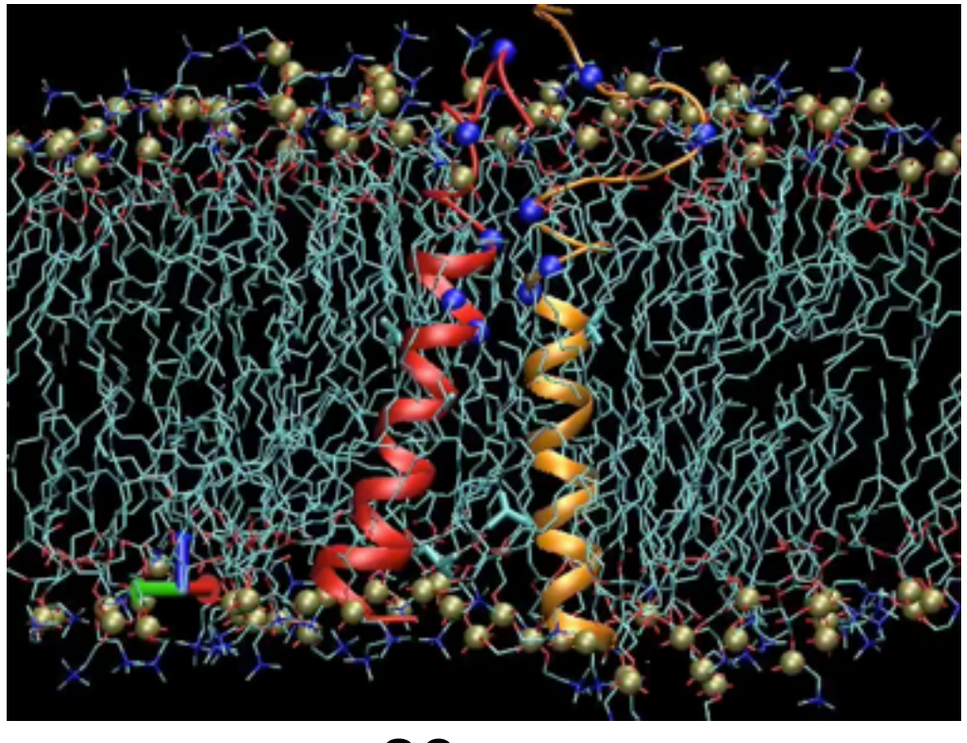
Naoyuki Miyashita, John E Straub, D Thirumalai, and Yuji Sugita,
Transmembrane structures of amyloid precursor protein dimer predicted by replica-exchange molecular dynamics simulations. *Journal of the American Chemical Society*, 131(10), pp.3438–3439 (2009)

32 replica



20 ns





30 ns

糖鎖分子の構造予測

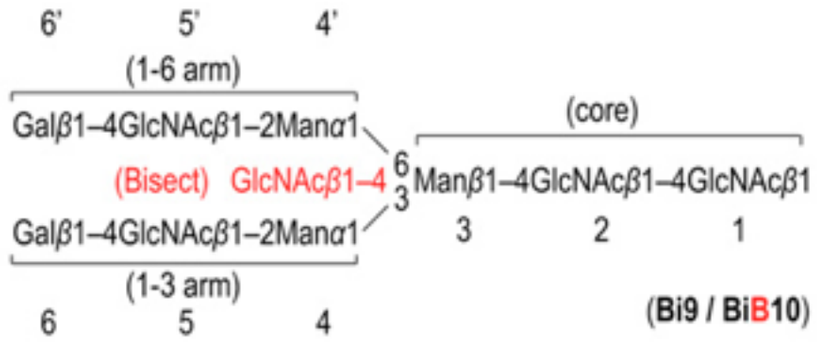
Suyong Re, Naoyuki Miyashita, Yoshiki Yamaguchi, Yuji Sugita, Structural Diversity and Changes in Conformational Equilibria of Biantennary Complex-Type N-Glycans in Water Revealed by Replica-Exchange Molecular Dynamics Simulation,

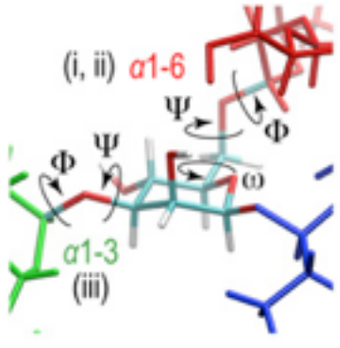
Biophys. J. **101**, Pages L44–L46 (2011)

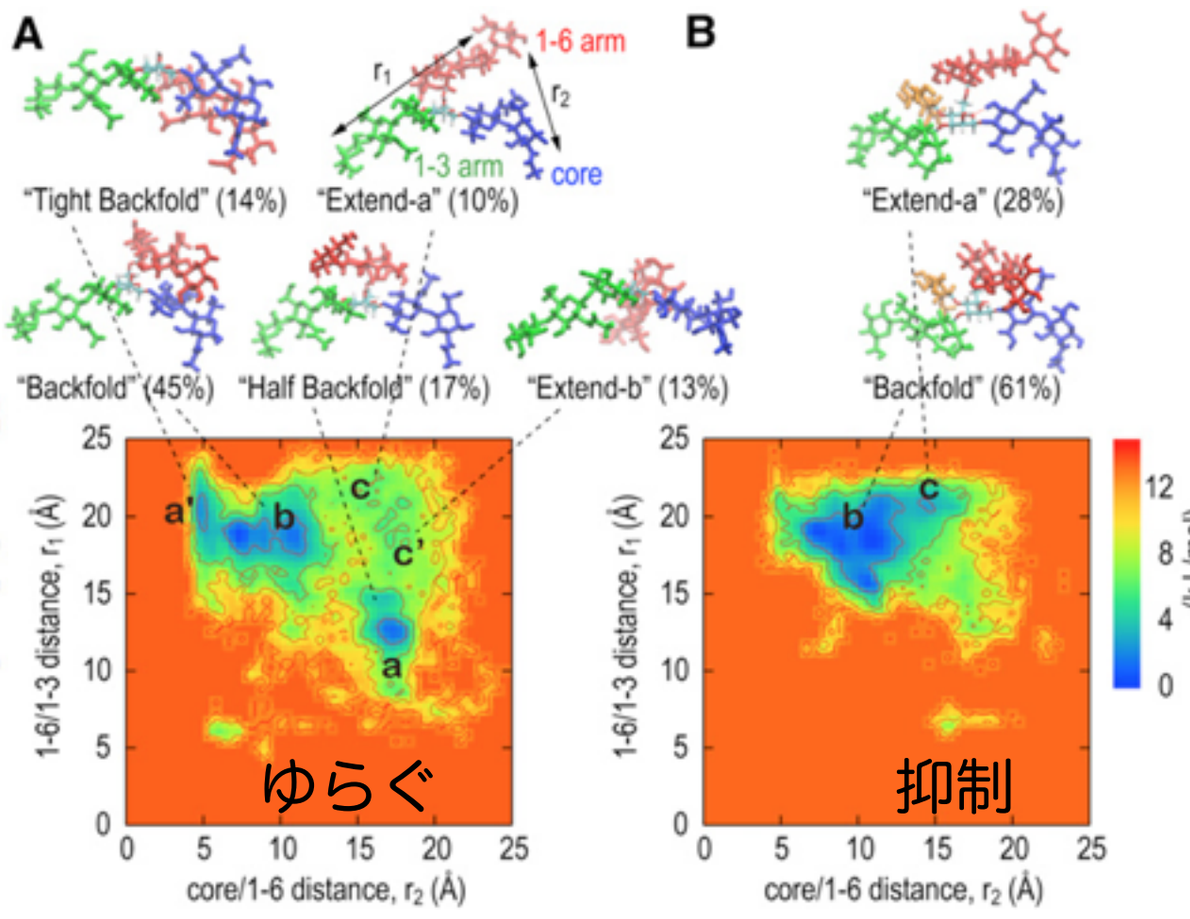
李博士(理研)

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。

90 replica







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

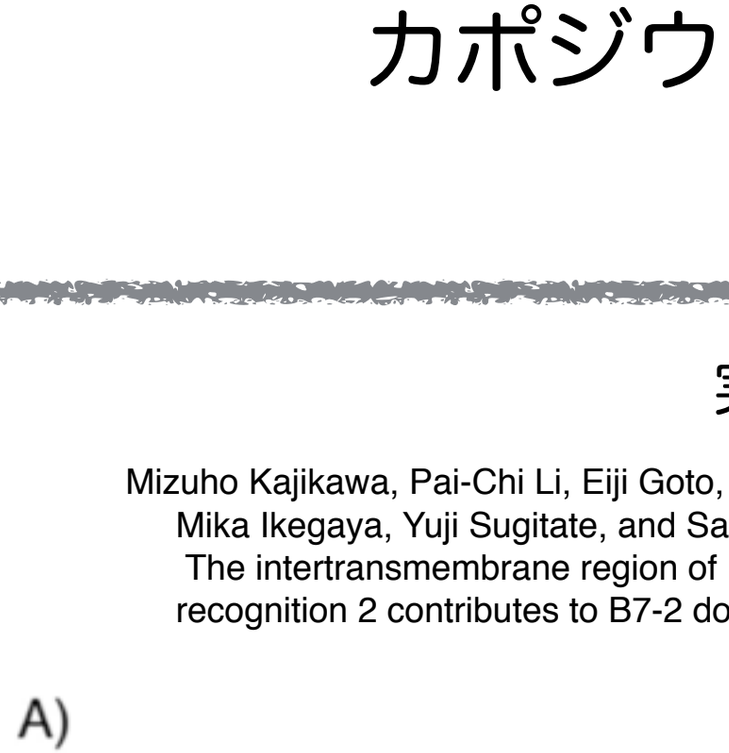
実験との共同研究

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)

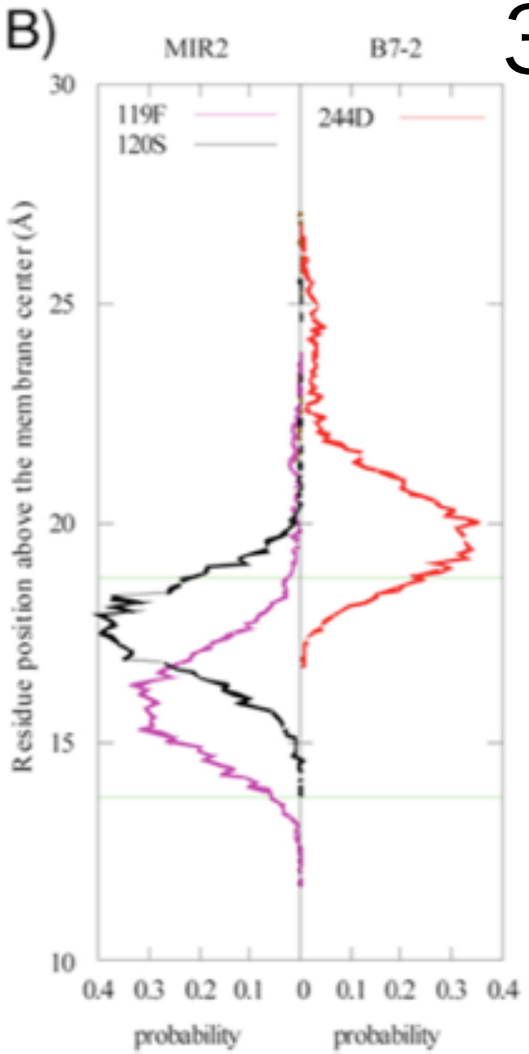
梶川博士(昭和薬大)
Pai-chi Lee博士（理研）

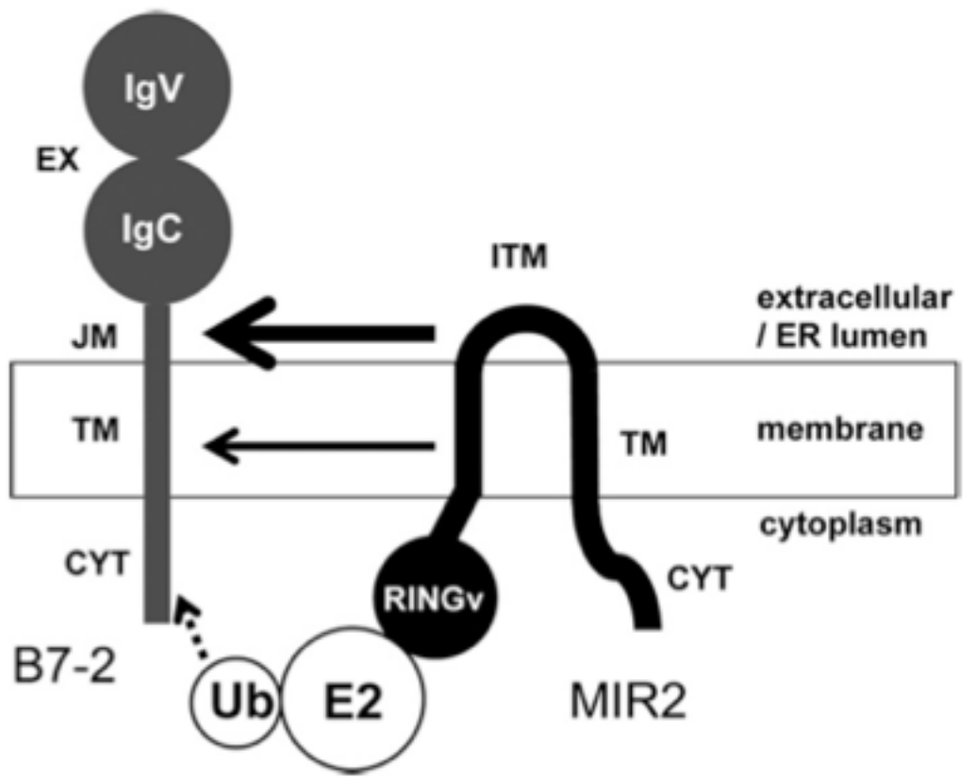
32 replica, 450 replica

A)



B)





4. REIN-Kの使い方

REINパッケージの構成

- rein
- supporting software
  - input_builder
  - batch_builder
  - remd_convert

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.sh 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.sh 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。

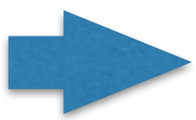
- $ vi auto_submitter.pl
- $ perl auto_submitter.pl

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..), Total, Kinetic,Potential, 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..), Total, Kinetic,Potential, 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

Author	Naoyuki Miyashita and Yuji Sugita
ライセンス	ライセンス：GPL version 3
引用	Naoyuki Miyashita, Suyong Re, and Yuji Sugita, “REIN: Replica-Exchange INterface for simulating protein dynamics and function.” International Journal of Quantum Chemistry, 115(5), 325–332. (2015). Naoyuki Miyashita, Yuji Sugita, M-2: Replica-exchange interface program (REIN), http://www.islim.org/islim-dl_e.html
Main developer & サポート	近畿大学　生物理工学部　宮下尚之
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）(2012-2014) Number 24700299.
  - 理研　計算機センター RIKEN Integrated Cluster of Clusters (RICC)
  - JSPS科研費　基盤研究（C）(2014-2016)　開発中