レプリカ交換分子動力学法を用いたシミュレーション並列化フレームワーク

伊藤正勝[†,††,☆] 長嶋雲兵[†,††]

生体分子のシミュレーションを開発する際には，正確な熱力学量を求めるための膨大な計算コストと，対象系に合わせて多様なシミュレーションプログラムを開発する煩雑さが問題となる．これに対し，我々は，レプリカ交換分子動力学法に基づいてシミュレーションを並列化し，計算時間を短縮するためのツールキット（REMD toolkit）を開発した．また，対象系に応じて，サンプリング方法，ポテンシャルエネルギー関数などの組合せを変えることができるように，ツールキットをカスタマイズ可能なソフトウェアコンポーネントの集まりとして設計した．これにより，様々なシミュレーション機能が，ツールキットが提供するコンポーネントと，外部プログラムに由来するコンポーネントの組合せとして実現される．ツールキットを検証するために，原子クラスタ Ar_{13}，オリゴペプチド $(Ala)_{10}$ といったモデルケースのそれぞれについて，プログラムを生成し，実行した．この結果，レプリカ数の増加により総計算量は圧縮され，さらに並列化によって計算時間は CPU 数に反比例して短縮されることが確認された．

A Composable Framework to Parallelize Simulation Programs through Replica-Exchange Molecular Dynamics

Masakatsu Ito[†,††,☆] and Umpei Nagashima[†,††]

We have developed a toolkit to generate a replica-exchange molecular dynamics program which accelerates the estimation of thermodynamical quantities. The toolkit is designed as a set of software components, so that any new variant of simulation program can be built by assembling suitable components. They are categorized according to three types of customizations : (1) parallelization of simulation programs, (2) selection of structure sampling method, and (3) incorporation of an arbitrary force field implementation into the program. The extensibility of the toolkit is demonstrated by generating new variants of replica-exchange molecular dynamics programs, and the efficiency of the generated programs is examined in the heat capacity estimation of Ar_{13} and $(Ala)_{10}$. It is shown that the replica-exchange scheme not only reduces the total computational cost with the increase in the number of replicas but achieves almost linear-speedup with the number of CPUs.

1. はじめに

近年のシミュレーション技法と分散コンピューティング技術の進歩により，分子シミュレーションのターゲットは単純な化学反応から複雑な生体プロセスへと広がりつつある．この結果，生体高分子の振舞いや熱力学量をシミュレーションに基づいて正確に評価することが，重要な課題となっている．特に，通常の分子動力学（MD）法では熱力学量評価に大きな誤差が生じがちであり，正確な値を得ようとすれば非現実的なほどに長い計算時間が必要となるため，こうした熱力学量評価のための計算量を圧縮できるようなシミュレーション技法を取り入れ，並列化によってさらに計算時間を短縮することが求められている．

これに対し，レプリカ交換分子動力学（REMD）法[11] を用いれば熱平衡分布を速やかに発生させることができ，熱力学量評価のための計算量は圧縮されると期待される．しかも，並列化に要する通信量と頻度が小さいため，Grid のような広域分散並列環境に適するという利点もある．しかし，薬物分子設計のような問題に適した REMD プログラムの開発が遅れてい

† 産業技術総合研究所グリッド研究センター
Grid Technology Research Center, National Institute of Advanced Industrial Science and Technology, AIST
†† 科学技術振興事業団計算科学技術活用型特定研究開発推進事業 “Research and Development for Applying Advanced Computational Science and Technology” of Japan Science and Technology Corporation
☆ 現在，株式会社富士通研究所ナノテクノロジー研究センター
Presently with Nanotechnology Research Center, FUJITSU LABORATORIES LTD.

るために，従来の MD パッケージを使わざるをえないことが多い．

つまり，問題の所在は，シミュレーションに関する理論からソフトウェアの領域へと移ってきている．理論的には，MD 法に温度交換スキームを追加するだけでただちに REMD アルゴリズムが導かれるが[11]，MD パッケージにコードを追加する際にはプログラミング上の問題に対処する必要がある．MD プログラムの中では，ポテンシャルエネルギーモデルが複雑なデータ構造として表現されているため，複雑さを封じ込める仕組みがなければ，MD プログラムを REMD プログラムに拡張し，並列化するために，煩雑なコーディングを避けることができない．

この問題に対しては，複雑なデータ構造をオブジェクト内にカプセル化し閉じ込めるというアプローチが有効である[7]．事実，NAMD [1] や ProteinDF [2] といったソフトウェアでは，オブジェクト指向型アプローチを採用し，生体高分子のトポロジと並列化メカニズムの複雑さを隠蔽することで，シミュレーション機能の拡張を促し，並列化効率の向上を図っている．NAMD では，C++類似のオブジェクト指向型言語 Charm++ [3] を用いて MD 法が実装され，Charm++オブジェクトの中に力場計算の詳細を隠蔽することで，物理化学的モデルの詳細に煩わされることなく動的負荷分散の仕組みを洗練させている．こうして，並列化メカニズムの拡張は続けられ，数百 CPU の規模で高いスケーラビリティが達成されている．ProteinDF は C++で密度汎関数法を実装しており，並列化にともなう複雑さをオブジェクトの内部に閉じ込めることで，高精度の電子状態計算をタンパク質のような巨大な分子に対しても適用可能としている．

ただし，MD プログラムを REMD プログラムへと拡張する際には，NAMD [1] や ProteinDF [2] のようにアプリケーションプログラムを 1 つだけ作れば十分というわけではない．問題によって REMD プログラムの中で呼び出すべきポテンシャルエネルギーのモデルが異なるため，様々なプログラムを生成する必要がある．こうした問題に対しては，オブジェクト指向型フレームワーク[8] が一般的な解決策となりうる．多くの分野でフレームワークが分野内で共通のアーキテクチャを提供することで，プログラム開発のためのコーディング量を最小化しているように，フレームワークとして REMD アルゴリズムを規定しておけば，問題固有のモデルを実装するだけで REMD プログラムを生成することができる．しかも，フレームワークのインタフェースに適合してさえいれば，モデルがどのように実装されていてもフレームワークに取り込むことができる．その実装は，既存の MD プログラム，あるいは，まったく新しいモデルを実現するために書き下したコードの，いずれであってもよい．このように，フレームワークがあれば任意のモデルに対応した REMD プログラムを生成できる．

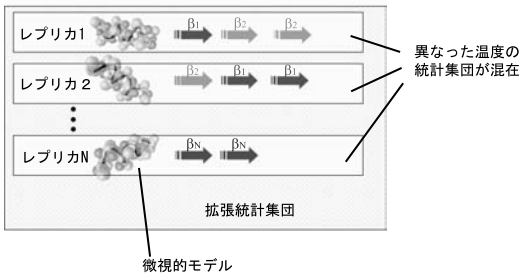

図 1 REMD シミュレーションにおける拡張統計集団の構成図
Fig. 1 Composite structure of an extended ensemble in an REMD simulation.

そこで，我々は分子シミュレーションプログラムに REMD アルゴリズムを追加し，分散並列環境で実行させるためのソフトウェアフレームワークとして，REMD toolkit [4] を開発した．本稿では，REMD toolkit を用いたシミュレーションプログラム並列化について報告する．2 章では，REMD アルゴリズムの概略とツールキットの設計について述べ，3 章で，ツールキットの実装について述べる．4 章では，フレームワークからシミュレーションプログラムを生成するための具体的手順と，得られたプログラムの実効性能について議論する．最後に，5 章で，まとめと残された課題について述べる．

2. REMD Toolkit の設計

2.1 レプリカ交換分子動力学法の概略

ツールキットの設計について述べる前に，微視的モデル，統計集団，拡張統計集団といった統計力学的概念について簡単に説明する．**図 1** はこれら 3 つの概念の関係を示す．この図にあるように，レプリカ交換分子動力学（REMD）法[11] におけるレプリカは，同じ種類の分子のコピーを意味している．

微視的モデル：原子座標，化学結合といった微視的状態から分子のポテンシャルエネルギーと原子に作用する力ベクトルを計算するために用いられるのが，微視的モデルである．MD 法ではこの力ベクトルに基づいて分子構造を少しずつ変化させている．

統計集団：生体プロセスの起こりやすさは生体分子の構造についての熱分布によって決定される．統計集団は熱分布に含まれる構造の集合であり，MD 法を用

いて数値的に発生させることができる．ここで，MD 法は構造サンプリング手法として使われていることになる．こうして得られた構造の分布が熱平衡分布に一致していれば，熱力学量を正確に求めることができる．しかし，ここで 2 つの問題がある．

1. 低温で生体分子が平衡状態に緩和するにはシミュレート可能な時間スケール（ピコ秒〜ナノ秒）に比べ，長い時間（> マイクロ秒）が必要であるため，MD 法のみでは正確な低温分布を生成できないことが多い．
2. MD シミュレーションから得られる熱力学量は，1 つの温度に対応した量に限られる．

拡張統計集団：これらの問題を解決するために，レプリカ交換分子動力学法シミュレーションでは，高温から低温まで，様々な温度 $\beta_1, \ldots, \beta_n$ の統計集団を集めて，拡張統計集団を構成する（図 1）．

それぞれの統計集団は，分子レプリカに対して，MD 法を適用することで生成される．これだけなら，莫大な計算時間が必要となってしまうが，REMD 法では所定のステップ数ごとにレプリカどうしで確率的に温度を交換することで低温 MD での平衡接近を加速する．これによって問題 1. は解決される．また，REMD 法ではマルチプルヒストグラム法[12] を使うことで，拡張統計集団に含まれる温度に加えて，任意の温度での熱力学量を計算することが可能となり，問題 2. も解決される．

以下にマルチプルヒストグラム法の概要を述べる．任意の温度で☆熱力学量 $\langle A \rangle_\beta$ を求めるためには，エネルギーごとの状態密度 $\Omega(E)$ が分かっていればよい．熱力学量は対応する物理量 $A(E)$ の $\Omega(E)$ で重みづけられた平均として，次式のように計算される．

$$\langle A \rangle_\beta = \frac{\sum_j A(E_j)\Omega(E_j)e^{-\beta E_j}}{\sum_j \Omega(E_j)e^{-\beta E_j}} \qquad (1)$$

しかし，通常の MD シミュレーションにおけるエネルギー E の出現回数 $n_i(E)$ から $\Omega(E)$ を求めると，ほとんどのエネルギーについて誤差が大きくなってしまう．

これに対し，REMD シミュレーションのように，N 個の温度について別々に $n_i(E)$ を数え上げていれば，$n_i(E)$ ごとの重みを調整することで誤差を小さくできる．誤差が最小化された $\Omega(E)$ は次の 2 つの式で与えられる．

☆ 式 (1)，(2) において，β は逆転温度であり，$1/(k\beta)$ が温度である．

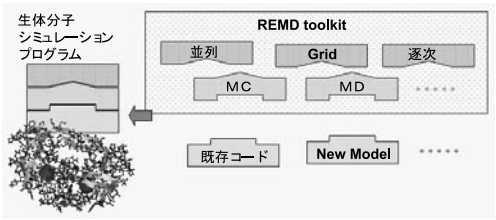

図 2 シミュレーションプログラムを生成する枠組みとしての REMD toolkit．ツールキットが提供するコンポーネントと外部のコンポーネントを組み合せることで，シミュレーションプログラムが生成される．

Fig. 2 A toolkit implementing a parallelized replica-exchange molecular dynamics was developed and then used as a software framework to generate variants of simulation programs by assembling the toolkit components and force field programs.

$$\Omega(E_j) = \frac{\sum_i^N g_i^{-1} n_i(E_j)}{\sum_i^N g_i^{-1} e^{f_i - \beta_i E_j}},$$
$$e^{-f_i} = \sum_j \Omega(E_j) e^{-\beta_i E_j} \qquad (2)$$

ここで，$g_i = 1 + 2\tau_i$ で，τ_i は i 番目の温度における相関時間である．マルチプルヒストグラム法では，これらを反復的に解くことで $\Omega(E)$ を決め，熱力学量 $\langle A \rangle_\beta$ を温度の関数として計算する．

2.2 設計要件

分子シミュレーションプログラムを REMD toolkit によって並列化するために，ツールキットは以下のような 2 つの設計要件を満たすべきである．

a. 様々な微視的モデルに対応できなければならない：微視的モデルは，対象系に応じて，スコアリング関数，AMBER[5]，CHARMM[6] といった分子力場，分子軌道法など多岐にわたる．そこで，ツールキットをオブジェクト指向型フレームワークとして設計した．本稿では，「フレームワーク」を，Gamma らの「デザインパターン」[8] における記述に従って，特定の分野でのアプリケーション生成を支援し，アーキテクチャなどの設計をあらかじめ定義するソフトウェア，の意味で用いる．フレームワークでは設計の再利用が強調される．一般的なフレームワークと同様，REMD toolkit そのものは実行可能プログラムではないが，このツールキットに開発者が微視的モデルに関するコードを組み込むたびに，新たなモデルに対応した REMD シミュレーションプログラムが生成される．

b. 並列化メカニズム，構造サンプリング手法，微視的モデルがそれぞれ独立にカスタマイズ可能でなければならない：たとえば，ある微視的モデルに対して，ツールキットが提供するすべての構造サンプリング手

法が利用可能であるべきである．並列化メカニズムについても同様である．そこで，REMD プログラムは**図 2** のように，ツールキットの中から選びだされた 2 つのコンポーネントと，外部から取り込む 1 つのコンポーネントから構成されるものとした．これらのコンポーネントの独立性を高めるために，コンポーネント間のデータのやりとりはオブジェクト指向型インタフェースを介するように制限した．

これにより，たとえば，並列化に関するコンポーネントを，Grid，PC クラスタ（MPI），逐次型などの環境に応じて変えたとしても，他のコンポーネントには影響が及ばず，カスタマイズ可能性が保証される☆．

また，分子シミュレーションプログラムからは，ポテンシャルエネルギーを計算するコードを抜き出し，オブジェクト指向型インタフェースを被せてコンポーネント化することで，既存プログラムの内部的実装が他のコンポーネントに影響を及ぼさないようにしている．

3. REMD Toolkit の実装

前章での設計要件を満たすように，クラス階層を構築し，オブジェクトを関連づけた．

3.1 拡張統計集団コンポーネント

拡張統計集団コンポーネントの UML クラス図を**図 3** に示す．下の方には他のオブジェクトの部品となるオブジェクトを配置し，上の方には部品オブジェクトを所有する複合オブジェクトを配置している．最上部にある *AbstReplicaSimulator* クラスは，全体的なコントロールを受け持つシミュレータオブジェクトの振舞いを宣言している．このオブジェクトがすべての部品オブジェクトを生成し，REMD シミュレーションを実行し，最後に部品オブジェクトの後片付けを行う．シミュレータオブジェクト自身は *main()* 関数によってインスタンス化される．ただし，*AbstReplicaSimulator* は抽象クラスなので，そのサブクラスのいずれかがインスタンス化される．*ReplicaSimulator* インスタンスはシミュレーションを逐次実行し，*MasterReplicaSimulator* は並列実行におけるマスタとして振る舞い，*WorkerReplicaSimulator* はワーカとなる．

図 3 の中ほどに示した *AbstEnsemble* と *ExtendedEnsemble* クラスは，シミュレータオブジェクトの部品オブジェクトの振舞いを規定している．これらのオブジェクトが，以下の 2 つのステップを繰り返すことで，レプリカ交換法シミュレーションは実現される．

(1) *AbstEnsemble* オブジェクトは平行かつ独立に，MD 計算（あるいは MC 計算）を所定のステップだけ行い，得られたエネルギーを *ExtendedEnsemble* オブジェクトに送る．

(2) *ExtendedEnsemble* はエネルギーを集め，熱分布を壊さないような確率で，分子レプリカの温度を交換し，新しい温度を統計集団オブジェクトに送り返す．

図 3 の最下部に示されているのは，統計管理オブジェクトのためのクラス群である．ここでは，熱力学量評価と統計集団オブジェクトと *ExtendedEnsemble* の間でのメッセージ転送が定義されている．*StatManager* クラスには，すべての統計集団オブジェクト（*AbstEnsemble*）からはエネルギーが，*ExtendedEnsemble* からは温度が送られ，これらの情報に基づいてマルチプルヒストグラム法が行われる．

ただし，並列実行の際には，マスタ，ワーカに分散された *ExtendedEnsemble* と統計集団オブジェクトの間でメッセージ転送を行うため，*StatManager* に加えて，そのプロキシ，スケルトンオブジェクト[8),9)] が生

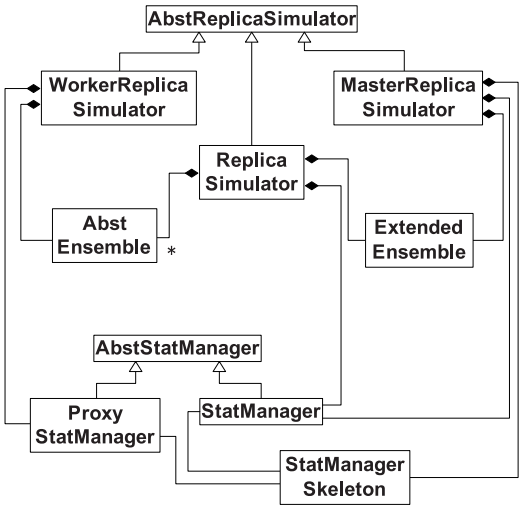

図 3 UML 記法に基づく，拡張統計集団のクラス図．矢印は継承関係，終端が菱形の実線は所有関係，実線は関連を示す．所有関係の終端にある * は複数のインスタンスが所有されていることを意味する．インスタンスの多重度が 1 である場合は終端の記号を省略した．

Fig. 3 UML class diagram for the extended ensemble component. Arrows indicate the generalization relationships, lines with diamond-ends indicate the composition relationships, and plain lines indicate the association relationships. * symbol indicates that several instances are created.

☆ 構造サンプリングに関しては MD 法と MC 法のいずれかを選択できる．MD 法の代わりに MC 法を用いた場合，生成されるプログラムはレプリカ交換法（REM）[10)] のシミュレーションとなる．

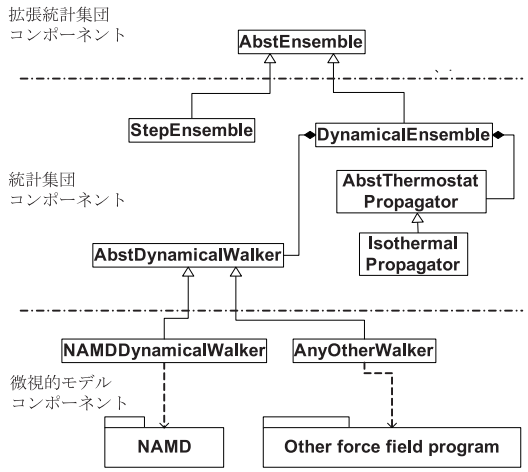

図 4 3 つのコンポーネントの関係を示す UML クラス図．拡張統計集団コンポーネント（図 3）と微視的モデルコンポーネントについては，統計集団コンポーネントに関連するクラスだけを示す．*StepEnsemble* 以外のモンテカルロ関連クラスは省略．タブのついたフォルダはソフトウェアパッケージを示す．クラスとソフトウェアパッケージの間を結ぶ破線は依存関係を示す．

Fig. 4 UML class diagram for the ensemble component and its related classes in the other two components. MC-related classes other than *StepEnsemble* are omitted. "Tabbed" shape indicates a software package. Dashed lines indicate dependency.

成される．これらのプロキシ，スケルトンオブジェクトの内部には，並列化ミドルウェアとして用いられている MPI に特有のコードがカプセル化され隠蔽されている．レプリカ交換法は微視的モデルにかかる計算コストに比べて，レプリカ間の通信はデータ量，頻度ともに小さく，並列化向きのシミュレーション方法であるが，さらに，我々の実装では通信量を最小化するために，レプリカ間で座標を交換させるのではなく，温度を交換している．

3.2 統計集団コンポーネント

AbstEnsemble のサブクラスは統計集団コンポーネントで定義されている（**図 4**）．REMD シミュレーションでは，*DynamicalEnsemble* がインスタンス化され，MD 法による構造サンプリング手法が提供される．これに対し，*StepEnsemble* がインスタンス化された場合は，MC 法による構造サンプリング手法が提供され，REMD ではなくレプリカ交換法（REM）シミュレーションが実現される．

DynamicalEnsemble の部品クラスとなっている *AbstDynamicalWalker* では，MD ステップの中から微視的モデルを呼び出すためのインタフェースを宣言している．

3.3 微視的モデルコンポーネント

ここでは，REMD toolkit のソフトウェアフレームワークとしての側面が強調されている．これまでの 2 つのコンポーネントでは，あらかじめ用意されたサブクラスの中で，インスタンス化されるクラスを切り替えるだけで，開発者がシミュレーションをカスタマイズできた．これに対し，このコンポーネントでは，対象系に適合したシミュレーションプログラムを生成するために，開発者がモデル固有のサブクラスを実装することを想定している．

実際の実装は微視的モデルコンポーネントで定義される．サブクラスを実装するには，以下に示す *AbstDynamicalWalker* 型オブジェクトのメソッドをオーバライドしさえすればよい（この実装を容易にするために，分子構造を表現するための標準的方法が，*AbstDynamicalWalker* のサブクラスとしてあらかじめ用意してある．そこで，これらのクラスを拡張してサブクラスを定義する場合は，さらにコーディングの量を減らすことができる）．

```
class AbstDynamicalWalker {
 public:
  AbstDynamicalWalker();
  virtual ~AbstDynamicalWalker() {}

  // 初期化
  virtual void init() {}
  // ポテンシャルエネルギーを返す
  virtual double map();
  // 新たな分子構造を発生
  virtual void
  walk(AbstThermostatPropagator& prop);

  // その他のメソッドとインスタンス変数は省略.
}
```

サブクラスでオーバライドされたメソッドは，統計集団コンポーネントから呼び出される．このコンポーネントからは，サブクラスの型の違いは見えず，*AbstDynamicalWalker* 型とだけ認識される．したがって，新たなサブクラスを追加する際に，拡張統計集団コンポーネントと統計集団コンポーネントのコードを修正する必要はない．

AbstDynamicalWalker のサブクラスを実装する際には，以下の 3 つの手順に沿って行う．

- **インスタンス変数：**構造，力ベクトルなど，分子の微視的状態を表現する変数をインスタンス変数とする．このデータ構造はクラスの外からは見えな

いため，開発者が自由に決めることができる．これらの変数は，コンストラクタと *init()* メソッドで初期化されるようにする．デストラクタは，インスタンス変数のメモリ解放といった終了処理を行う．

- ***map()* メソッド：**エネルギーと力ベクトルを計算するコードを書き（あるいは，既存コードを再利用し），エネルギーをこのメソッドの返り値とする．
- ***walk()* メソッド：**ここには，統計集団コンポーネントから定温 MD を行うオブジェクト *prop* が渡されるので，原子座標と力ベクトルを *prop* に渡して，新たな原子座標が計算されるように実装する．

4. シミュレーションプログラムの生成と実行

異なる性質を備えた 2 つの対象系のそれぞれについて，別々の微視的モデルに対応するシミュレーションプログラムを生成した．前者は，ツールキットが正常に動作することを確認（4.2 節）するために，後者は実行効率を検証（4.3 節）するために用いた．

フレームワークに基づいてプログラムを生成する場合，メインループから呼び出される側のサブクラスを書き，フレームワークのコードとともにコンパイルして実行ファイルを得る[8]．メインループの実装はフレームワークが提供するので，開発者は 3.3 節で示した手順でサブクラスを定義しさえすれば，シミュレーションプログラムを得ることができる．

4.1 微視的モデルに対応したプログラムの生成

新規コードの取り込み：まず，単純な対象系，アルゴンクラスタ Ar_{13}，に特化した REMD プログラムを生成するために，原子クラスタ固有のコードを新たに書き加え，*AbstDynamicalWalker* のサブクラスとした．

原子座標と力ベクトルを単純な配列で表現し，*map()* 以外のメソッドは，3.3 節のように実装した．しかし，*map()* メソッドは，原子クラスタは有限温度で分解し，平均エネルギーが解離極限に一致してしまうという問題に対処する必要があった．そこで，我々は，原子の解離を抑制するために，Lennard-Jones ポテンシャルに原子を拘束するための項を付け加えた．

ここでシミュレーションプログラムを生成するために，新たに書き加えたコードは，サブクラス，*main()* 関数を合わせて，140 行ほど☆であり，これに対して，再利用された REMD toolkit のコードは 2 万行であった．しかも，再利用するために，ソースコード全体に目を通す必要はなく，3.3 節で示したように一部のクラス宣言を理解すればよい．

既存の MD プログラムの取り込み：次に，微視的モデルとして広く用いられている，CHARMM 力場[6] に対応した REMD プログラムを生成するために，*AbstDynamicalWalker* のサブクラスから *NAMDDynamicalWalker* クラスを派生させた．

NAMDDynamicalWalker を実装するために，NAMD の逐次版である Mindy プログラム[1] を利用し，ポテンシャルエネルギーや力場ベクトルを求めるための，複雑な処理はすべて Mindy に委譲した．*NAMDDynamicalWalker* クラスで実装したコードは，NAMD オブジェクトのインタフェースを *AbstDynamicalWalker* のインタフェースに変換している．

まず，*NAMDDynamicalWalker* を定義するために，6 つの NAMD オブジェクトをインスタンス変数とし，原子座標，速度ベクトル，力ベクトルを表すインスタンス変数は，NAMD オブジェクトのメソッドの引数型に合わせた．コンストラクタでは，Mindy の入力ファイルの名前を NAMD オブジェクトに渡して初期化を行い，NAMD オブジェクトから原子座標の初期値を受け取った．*map()* メソッドは原子座標を NAMD オブジェクトに渡し，エネルギーと力ベクトルを受け取るだけとした．

ここでシミュレーションプログラムを生成するために，新たに書き加えたコードは，330 行ほどであり，これに対して，再利用されたコードは Mindy，REMD toolkit で合わせて約 3 万行となった．また，ツールキットを再利用するためにソースコード全体を理解する必要がないのと同様に，Mindy では 6 つの NAMD オブジェクトのクラス宣言を参照するだけで，既存のコードを再利用できた（一般に，既存の MD コードを再利用する際には，モジュールとして微視的モデルのコードを括り出す段階に，最も時間がかかると思われる．Mindy では微視的モデル（CHARMM 力場）が 2 つの C++クラス，ComputeBonded と ComputeNonbonded にまとめられていたため括り出しをする必要がなかったが，GROMACS[13] を取り込む際には，C のソースコードを修正する必要があった）．

これらの例で示されるように，REMD toolkit は，対象系に合わせて適切なモデルの実装を取り込めるという実用性とともに，ツールキットのコードを修正することなく使えるという再利用性を備えている．

4.2 比熱による REMD toolkit の動作検証

アルゴンクラスタ Ar_{13} に特化したプログラムを用いて，REMD シミュレーションを行い，Davis，Jellinek，

☆ コメント行，空行，プリプロセッサへの指示行を除く．

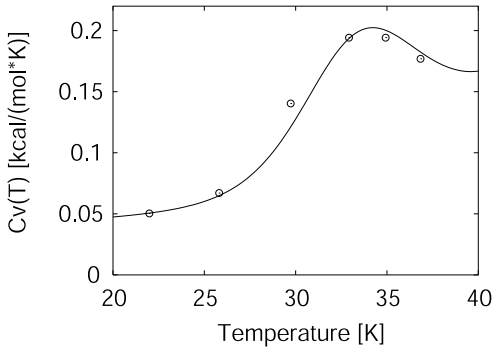

図 5 Ar_{13} の比熱を温度に対するプロット．実線は REMD シミュレーションから得られた．白丸は文献 14) の図 3 からスキャンした．

Fig. 5 Heat capacity of Ar_{13} as a function of temperature. Solid curve was obtained from the REMD simulation, and open circles are from Fig. 3 of Ref. 14).

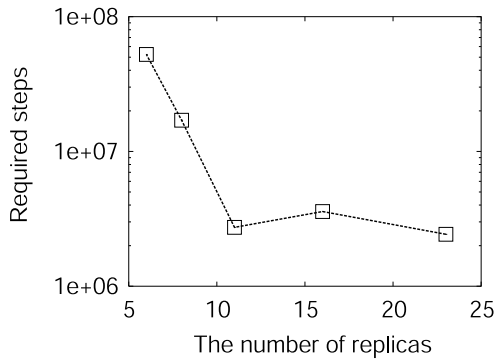

図 6 比熱誤差を 0.01 kcal/mol · K 以下にするために必要なシミュレーション ステップ数のレプリカ数に対するプロット

Fig. 6 Number of required conformations to decrease the error of heat capacity to less than 0.01 kcal/mol · K is shown as a function of the number of replicas.

Berry によって行われた MC シミュレーション[14] と比較することで，REMD toolkit の動作を検証した．両方の計算で，使用したモデルは Lennard-Jones ポテンシャルに基づいているため，平衡状態の系の性質は一致するはずである．特に，アルゴンクラスタは固相・液相の両方で存在し，2 つの相の間の転移は比熱の温度依存性に現れることが知られているため，平衡系の系の性質として比熱を用いれば，その温度依存性によって生成されたプログラムの正常動作を確認できる．また，微視的モデルは同じであっても，Berry らのプログラムは MC シミュレーションを行い，我々の REMD toolkit は，REMD シミュレーションを行っているため，REMD アルゴリズムが我々のツールキットに正しく実装されているかを検証できる．

そこで，我々は，REMD シミュレーションを用いて比熱を計算し，Davis らの結果と比較した．比熱計算の収束を速めるために，($N =$) 12 個のレプリカを用い，温度は $T_L = 20.00\,\mathrm{K}$ から $T_H = 42.87\,\mathrm{K}$ まで式 (3) に基づいて指数関数的に変化させた．

$$T_i = T_L \times \left(\frac{T_H}{T_L}\right)^{\frac{i}{N-1}}, i = 0, \ldots, N \qquad (3)$$

マルチプルヒストグラム法[12] によって計算した比熱の温度依存性を**図 5** に示す．我々の結果は Davis らの結果とほとんど一致しており，26 K を超えると比熱は急激に増加し，34 K でピークとなる．鋭いピークではなく，幅の広いピークが得られたことは，アルゴンクラスタの大きさが有限であり，マクロ系のように相転移が急激ではないために，比熱が発散しないことを反映している．

4.3 実行性能

CHARMM 力場[6] に対応したプログラムを用いて，オリゴペプチド $(Ala)_{10}$ の REMD シミュレーションを行った．ここで得られる比熱評価の精度向上，あるいは，計算時間の短縮といった効果は，微視的モデルの変更にともなうことではなく，MD アルゴリズムを REMD アルゴリズムへと拡張することによって生じている．そして，REMD アルゴリズムはツールキットによって実装されているため，これらの結果はツールキットそのものに帰することができ，微視的モデルを実装するために他のコードを取り込んだ場合にも，同じような効果が得られると期待できる．

力場パラメーターは CHARMM19 を用い，32 個のレプリカ，温度は $T_L = 174.12\,\mathrm{K}$ から $T_H = 800\,\mathrm{K}$ まで式 (3) に基づいて変化させた．さらに，レプリカ数を 6，8，11，16，23 のように変えて，REMD シミュレーションを行い，比熱は 250 K から 800 K まで，マルチプルヒストグラム法[12] によって求め，レプリカ数 32 のシミュレーションとの RMSD（root mean square deviation）を誤差と見なした．

図 6 でプロットした所要計算量（required steps）は，比熱誤差が 0.01 kcal/mol · K 以下になるまでに，REMD シミュレーション全体で MD 計算に費やされたステップ数の合計である．所要計算量 r と逐次型シミュレーションにおける計算時間 t_s は，

$$t_s = r\,t_m \qquad (4)$$

のように比例している．ここで，t_m は MD の 1 ステップを進めるために必要な時間で，微視的モデルに基づいてポテンシャルエネルギーを計算する時間にほぼ等しい．

所要計算量はレプリカ数の増大とともに，初めは急

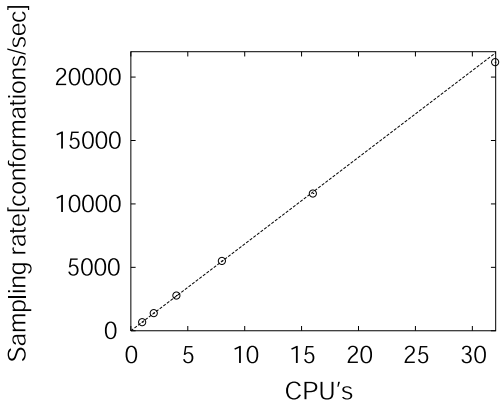

図 7 CPU 数の関数として分子構造の標本化周波数をプロットした．白丸は測定結果，破線は逐次版の速度を外挿した（勾配 683.8 [conformations/sec]）．

Fig. 7 The sampling rate as a function of the number of CPUs. The open circles are the measured sampling rates. The dashed line is extrapolated from the sampling rate of the serial version; Its slope is 638.8 [conformations/sec · CPU].

激に減少している．つまり，並列化をする前の段階で，レプリカ数を単純に増やすだけでも計算時間は急激に短縮される．やがて，レプリカ数が 8 を超えると，同程度の計算量に落ち着く傾向が認められ，並列化をすることでさらなる計算時間の短縮が期待できる．

ところで，MD シミュレーションはレプリカ数が 1 であるような REMD シミュレーションと見なすことができる．そこで，図 6 の横軸を左に延長した先のレプリカ数 1 の場合が，MD シミュレーションを熱力学量評価に用いた場合に相当する．この場合，MD シミュレーションで指定された温度については，正確な熱力学量を求めることができるが，そこから外れた温度については熱力学量を求めることができない．このように，レプリカ数の増加にともなって所要計算量が急激に減少するという傾向は，レプリカ数 1 の場合（MD シミュレーション）から始まっていると考えられる．

並列化性能は 32 個のレプリカからなる REMD シミュレーションを，16 ノードの PC Linux クラスタ（ノードごとに dual Pentium III 1400 MHz，メモリ 2.3 GB）で実行することで評価した．**図 7** の縦軸（sampling rate）は，一秒に計算される MD ステップ数をすべてのレプリカにわたって合計した値である．シミュレーションでは MD ステップごとに新たな構造が生成されるため，この値は 1 つの分子構造を標本化するのにかかる時間の逆数，標本化周波数に等しい．熱力学量の統計的誤差は標本の数に依存しているため，標本化周波数が大きければそれだけ早く所定の誤差範囲で熱力学量が得られることになる．

図 7 に示すように，並列化されたシミュレーションの標本化周波数は CPU 数に対してほぼ線形に増加し，32 CPU で非並列版の 0.97×32 倍の加速が得られた．レプリカ交換法は微視的モデルにかかる計算時間に比べて，レプリカ間の通信はデータ量，頻度ともに小さく，並列化向きのシミュレーション方法である．対象となる分子が大きくなるにつれ，計算時間は長くなるが，交換するのは温度だけであるため，分子サイズの増大とともに，さらに効率が向上することが期待される．

レプリカ数 N のシミュレーションでは，CPU それぞれに 1 つのレプリカが割り振られたとき，つまり，並列度 N が限界である．そこで，レプリカ数 N を所要計算量 r が定常になるところまで増やし，通信のオーバヘッドが無視できる程度に大きな分子を，最大の並列度でシミュレートしたとき，計算時間 t_p は次式のように短縮される．

$$t_p = \frac{r\, t_m}{N} \tag{5}$$

ただし，レプリカ数を増加させる上限が存在するため，並列化による計算時間短縮にも限界がある．シミュレーション全体の計算量を固定してレプリカ数を増加させれば，レプリカごとにシミュレートされる時間は減少していくことになるが，一方で，レプリカ間の統計的独立性を保つためにシミュレートされる時間は系の緩和時間にくらべて十分に長い必要がある．したがって，レプリカごとのシミュレーション時間は緩和時間より短くすることはできない．この上限を超えてレプリカ数を増やしていくと，所要計算量はレプリカ数に比例して増えていくと考えられる．しかし，緩和時間の長さが問題になるのは巨大なタンパク質分子のように振動エネルギー緩和が遅い系に限られる．実際，$(\mathrm{Ala})_{10}$ のシミュレーションでは総計算量が増加に転じる傾向は見られなかった．

5. まとめと今後の課題

通常の MC，MD シミュレーションでは誤差を抑えることの困難な熱力学量を評価するために，REMD 法を実装した REMD toolkit を開発した．REMD アルゴリズムのカスタマイズ可能性と，問題固有の微視的モデルとの組合せによる利点を最大限に引き出すために，我々のツールキットをオブジェクト指向型フレームワークとして設計し，独立性の高いコンポーネントの集まりとして実装した．

ツールキットの拡張性を実証するために，原子クラスタのための力場と CHARMM 力場のそれぞれに対

応したシミュレーションプログラムを生成させた．実行性能を検証するために，$(Ala)_{10}$ の比熱誤差とレプリカ数の関係，CPU 数の増加にともなう構造サンプリングのスピードアップを求めた．その結果，レプリカ数の増大とともに，総計算コストは減少し，さらに並列化を行うと，CPU 数に対してほぼ線形に分子構造の標本化周波数が向上した．

現在，レセプタータンパク・阻害材の会合過程などのより大規模なシミュレーションを行うべく，

- Grid 化された拡張統計集団コンポーネント[15]，
- GROMACS[13] に対応した微視的モデルコンポーネント，

を組み合せて，シミュレーションプログラムの機能拡張を進めている．GROMACS は力場計算の効率化を図るとともに，Pentium IV のアセンブリ言語によって内部ループを高速化しており[13]，REMD toolkit による高速化と合わせて，さらに，REMD プログラムを高速化すると期待される．

REMD toolkit のソースコードは GPL のもとで公開している[16]．

謝辞 本研究は科学技術振興事業団の行う計算科学技術活用型特定研究開発推進事業による助成を受けて行われた．

参 考 文 献

1) Kale, L., Skeel, R., Bhandarkar, M., Brunner, R., Grusoy, A., Karawets, N., Phillips, J., Shinozaki, A., Varadarajan, K. and Schulten, K.: NAMD2: Greater scalability for parallel molecular dynamics, *J. Comp. Phys.*, Vol.151, pp.283–312 (1999).（Mindy は NAMD のソースコードを単純化したプログラム．http://www.ks.uiuc.edu/Development/MDTools/mindy/）
2) Sato, F., Yoshihiro, T., Era, M. and Kashiwagi, H.: Calculation of all-electron wavefunction of hemoprotein cytochrome c by density functional theory, *Chem. Phys. Lett.*, Vol.341, pp.645–651 (2001).
3) Kale, L.V. and Krishnan, S.: CHARM++: A Portable Concurrent Object Oriented System Based On C++, *Proc. Conference on Object Oriented Programming Systems, Languages and Applications*, Sept.-Oct. 1993, ACM Sigplan Notes, Vol.28, No.10, pp.91–108 (1993).
4) Ito, M., Nishikawa, T. and Nagashima, U.: Replica-exchange molecular dynamics toolkit: a software framework to parallelize simualtions, *J. Comp. Chem.*, submitted.
5) Weiner, P.K. amd Kollman, P.A.: AMBER: Assisted Model Building with Energy Refinement, A General Program for Modeling Molecules and Their Interactions, *J. Comp. Chem.*, Vol.2, pp.287–303 (1981).
6) Brooks, B.R., Bruccoleri, R.E., Olafson, B.D. and States, D.J.: CHARMM: A program for macromolecular energy, minmimization and dynamics calculations, *J. Comp. Chem.*, Vol.4, pp.187–217 (1983).
7) Barton, J.J. amd Nackman, L.R.: *Scientific and Engineering C++: an introduction with advanced techniques and examples*, Addison-Wesley Longman, Inc.(1994). Larman, C.（著），依田光江（訳），依田智生，今野　睦（監訳）：実践 UML パターンによるオブジェクト指向開発ガイド，ピアソン・エデュケーション（株）(1998).
8) Gamma, E., Helm, R., Johnson, R. and Vlissides, J.（著），本位田真一，吉田和樹（監訳）：オブジェクト指向における再利用のためのデザインパターン，ソフトバンク（株）(1997).
9) Lea, D.: *Concurrent Programming in Java*, 2nd Edition, Addison-Wesley Pearson Education (2000).
10) Hukushima, K. and Nemoto, K.: Exchange Monte Carlo Method and Application to Spin Glass Simulations, *J. Phys. Soc. Jpn.*, Vol.65, pp.1604–1608 (1996).
11) Sugita, Y. and Okamoto, Y.: Replica-exchange molecular dynamics method for protein folding, *Chem. Phys. Lett.*, Vol.314, pp.141–151 (1999).
12) Kumar, S., Bouzida, D., Swendsen, R.H., Kollman, P.A. and Rosenberg, J.M.: The Weighted Histgram Analysis Method for Free-Energy Calculation on Biomolecules, I. The Method, *J. Comp. Chem.*, Vol.13, pp.1011–1021 (1992).
13) Berendsen, H.J.C., van der Spoel, D. and van Drunen, R.: *Comp. Phys. Comm.*, Vol.91, pp.43–56 (1995).
14) Davis, H.L., Jellinek, J. and Berry, R.S.: Melting and freezing in isothermal $(Ar)_{13}$ clusters, *J. Chem. Phys.*, Vol.86, pp.6456–6464 (1987).
15) 佐藤　仁，伊藤正勝，中田秀基，松岡　聡：レプリカ交換分子動力学シミュレーター REMD Toolkit のグリッド上での実行，情報処理研究会報告 2003–HPC-95, pp.41–46 (2003).
16) REMD toolkit の開発バージョンは http://sourceforge.net/projects/remdtk の CVS レポジトリからダウンロード可能．

(平成 15 年 10 月 9 日受付)
(平成 16 年 1 月 21 日採録)

伊藤 正勝

昭和 43 年生．平成 2 年京都大学工学部合成化学科卒業．平成 4 年同大学大学院工学研究科分子工学専攻修士課程修了．平成 9 年総合研究大学院大学数物科学研究科構造分子科学専攻博士課程修了．同年分子科学研究所非常勤研究員，平成 9 年日本学術振興会未来開拓学術研究推進事業日本学術振興会研究員，平成 14 年科学技術振興事業団計算科学技術研究員を経て，平成 16 年（株）富士通研究所入社．博士（理学）．生体分子シミュレーションの研究と，そのためのソフトウェアフレームワークの開発に従事．日本生物物理学会，日本物理学会各会員．

長嶋 雲兵（正会員）

昭和 30 年生．昭和 58 年北海道大学大学院理学研究科博士後期課程化学第二専攻修了．理学博士．同年岡崎国立共同研究機構分子科学研究所電子計算機センター助手．平成 4 年お茶の水女子大学理学部情報科学科助教授，平成 8 年同教授，平成 10 年通産省工業技術院物質工学工業技術研究所基礎部理論化学研究室長．平成 11 年同産業技術融合領域研究所計算科学研究グループ長，平成 13 年独立行政法人産業技術総合研究所先端情報計算センター情報基盤研究開発室長，平成 14 年より同グリッド研究センター統括研究員．筑波大学連携大学院大学教授．計算化学，情報化学，大規模数値計算，広域分散並列処理の研究開発に従事．日本化学会，IEEE，応用数理学会，コンピュータ化学会各会員．