cosgene 機能追加
詳細設計書

2018 年 2 月 5 日

目次

１．目的と試作項目

将来の超並列計算に対するMDの実行速度向上方法として空間分割による並列実行があり、本作業では空間分割による並列計算を行うMDプログラムを試作する。

今回の試作では以下の３点の機能のプロトタイピングを行う。

(1)系の空間分割

現在の超並列計算機は数百台から数万台の分散メモリ型計算機で構成され、各計算機が並列でプロセスを実行、プロセス間のデータ交換はネットワーク通信で行う実装が主流である。

将来的なシミュレーションでは数百万以上の原子に対し、1000台以上の並列実行でのMDを行うことが想定される。

本試作では、系を構成する原子を座標空間で分割し、並列計算機の各構成要素にそれぞれの空間の原子のMD計算を割り当てるプロトタイプを作成する。

(2)空間分割でのプロセス間通信

空間分割では、相互作用計算のため隣接する空間の原子情報を相互に伝達する必要がある。また、分割した座標空間にまたがる原子の移動により構成原子の動的な変更を行う機能が必要となる。

本試作では、隣接する座標空間の原子の情報交換、および分割した座標空間にまたがる原子の移動をプロセス間通信で行うプロトタイプを作成する。

(3)空間分割した系での相互作用計算・積分器の並列実行

空間分割された系では、各原子の相互作用計算・積分器は独立に実行可能で、このことがMD計算の並列性の向上につながる。

本試作では、分割した空間の相互作用計算・積分器を並列に行うプロトタイプを作成する。

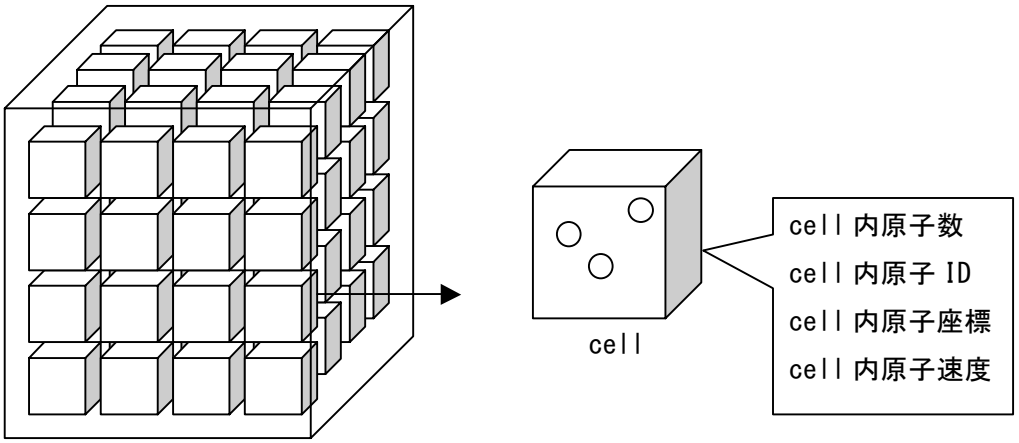
２．系の空間分割の実装方針

２．１．system と cell

本ドキュメントでは以降、系全体を"system"と定義し、系を分割した空間における最小の構成要素を"cell"と定義する。

１つの cell は下記の情報を持ち、すべての cell の情報を統合することで system の情報を表現する。

・cell の領域：cell の上下限座標、ID
・原子の座標と速度：cell 内に存在する原子の個数、および構成原子の ID・座標・速度



system と cell

cell は並列計算機での計算対象の最小単位であり、並列計算において 1cell は 1 プロセスに割り当てられる。

使用環境によっては最適な cell の分割数に対し、実際の CPU 数が不足することも想定されるため、1CPU への cell の割り当て数の上限は設けない。

cell 数が CPU 数よりも多い場合には、ラウンドロビン(巡回)方式でプロセスを CPU に割り当てることとする。

(例えば、8*8*8 のセルを 16 並列実行可能とする。この場合 1 プロセスが 4 つのセルの空間の MD を実行する)

試作環境では 2CPU で 4GB のメモリが上限であるため、下記のプログラム上の上限を持たせた。ただし、計算資源に応じてプログラムの定数部分を書き換えることで拡張可能とすることとした。

#	対象	試作の上限値	超並列計算機を想定した値
1	cell 数	64(4*4*4)	4096(16*16*16)
2	系の最大原子数	1,000,000	10,000,000
3	1 セル内の原子数	20,000	40,000
4	プロセス数	64(4*4*4)	4096(16*16*16)
5	1 cell の原子情報メモリ使用量	16.0MB	34.0MB

原子座標、力、速度および 1-5 相互作用リスト以外の情報は全プロセスで保持することし、2000 原子の蛋白質で bond=2000, angle=3000, torsion=6000, improper=400 と仮定した場合、500 万原子での基本的な MD での必要な情報量は以下のとおりである。

#	項目	データ	バイト数	個数(500 万原子)	使用メモリ
1	原子	mass	8	5,000,000	40MB
2		type	4	5,000,000	20MB
3		1-2, 3, 4 リスト	100	5,000,000	500MB
4		名前	4	5,000,000	20MB
5		所属残基 ID	4	5,000,000	20MB
6		所属チェイン ID	4	5,000,000	20MB
7	bond	ID	8	5,000,000	40MB
8		係数	8	5,000,000	40MB
9		平衡距離	8	5,000,000	40MB
10	angle	ID	12	7,500,000	90MB
11		係数	8	7,500,000	60MB
12		平衡角	8	7,500,000	60MB
13	torsion	原子 ID	16	15,000,000	240MB
14		分割数	4	15,000,000	60MB
15		初期角	8	15,000,000	120MB
16		係数	8	15,000,000	120MB
17		1-4 フラグ	4	15,000,000	60MB
18	improper	原子 ID	16	1,000,000	16MB
19		初期角	8	1,000,000	8MB
20	残基	先頭原子 ID	4	500,000	2MB
21		最終原子 ID	4	500,000	2MB
22		名前	4	500,000	2MB
23	分子	先頭原子 ID	4	10,000	0.04MB
24		最終原子 ID	4	10,000	0.04MB
25		名前	4	10,000	0.04MB
合計					1,580MB

1cel あたりの相互作用テーブルのメモリ使用量を 1000*原子数*4 バイトと仮定すると、20,000 原子の相互作用テーブルは 80MB である。

500 万原子系での 1 cell 当たりのメモリ使用量は 1,696MB と計算される。

２．２．cell の概要

cell は計算空間に含まれる原子の情報を保持し、cell が割り当てられた CPU は当該 cell に含まれる原子の相互作用計算、および積分器を実行する。

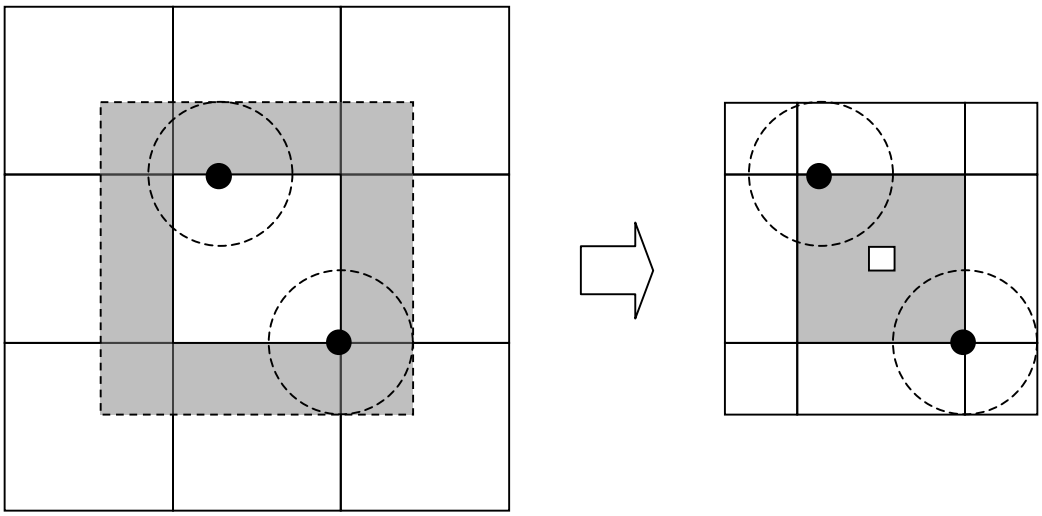
cell での相互作用計算では隣接するセルの原子との相互作用計算のため、当該 cell 原子と長距離相互作用が発生する可能性のある隣接 cell の矩形の空間(参照領域)の原子の ID と座標情報を持たせることとする。

参照領域オーバラップの範囲は当該セルからの距離がカットオフ距離以内となる空間で、この距離はシステムの入力値から設定する。

同時に cell 内には他の cell の参照領域となる可能性がある範囲が存在し、これをオーバラップ領域と呼ぶこととする。

参照領域とオーバラップ領域は、以下のように定義される。

参照領域：当該 cell が参照する他の cell の計算領域
オーバラップ領域：他の cell が参照する当該 cell の領域



cell の参照領域領域(左)とオーバラップ領域(右)

３次元の空間分割で相互作用計算のための原子座標の通信は、オーバラップ領域を参照領域に渡す計 26 個の cell に対する送信が必要となる。

２．３．空間分割でのプロセス間通信

(1) プロセス間通信

空間分割では、相互作用計算のため隣接する空間の原子情報の交換、および分割した空間にまたがる原子の移動を実現するためのプロセス間通信を行う。

本試作では、プロセス間通信は MPI 通信を使用することとする。、また、プロセス数が多い場合の１対多、多対多の通信は通信量が爆発する可能性があるため、１対１通信を優先的に使用することとする。

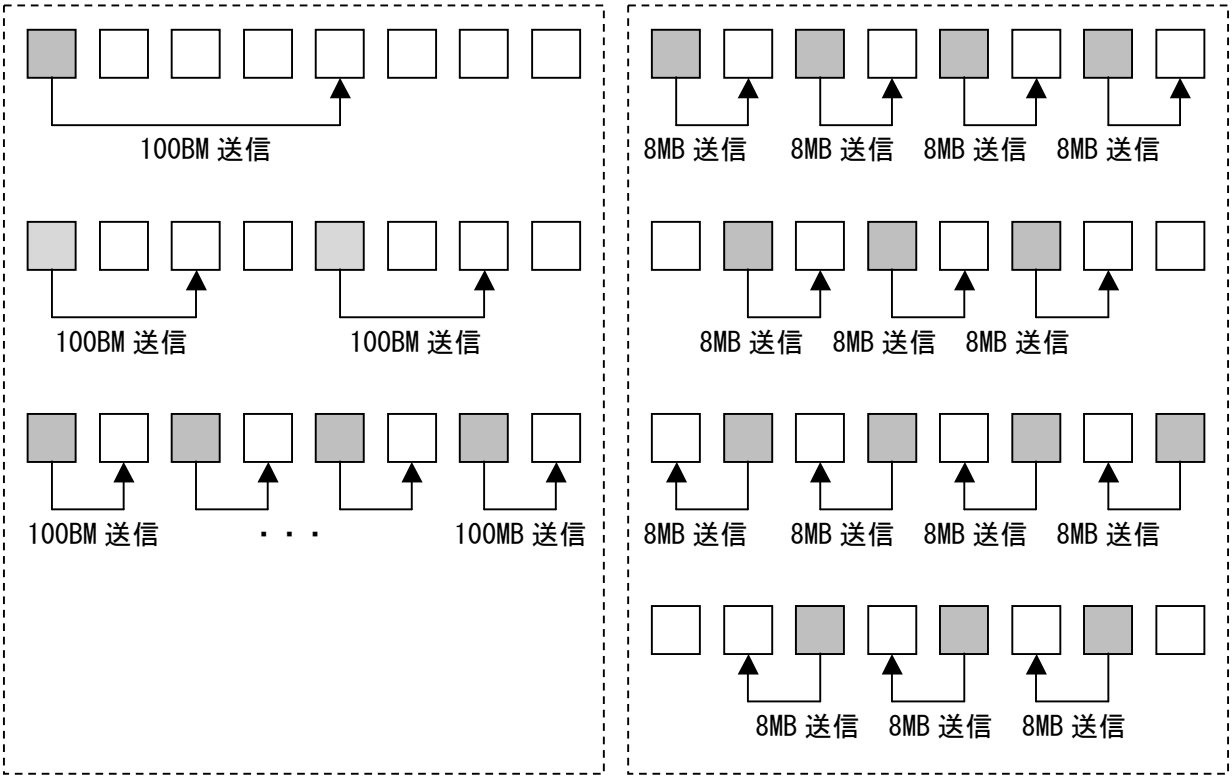
前章で述べたように隣接する cell 間では移動する原子の情報の通信、および参照領域の原子座標の通信が必要で、1 つの cell に対し 26 個の cell との通信が必要である。

空間分割した cell では隣接する cell に対し参照部分の原子情報のみを渡すことになる。

また、2 つ以上離れた cell 同士の通信は同時に実行可能であるため、同時に複数のノード間通信が可能な並列計算機においては、従来の同期型の１対多通信に対し実行時間に対する通信コストの大幅な削減が期待できる。

相互作用計算のための座標の通信方法として、従来の broadcast 通信と今回の試作での send/recv 型通信で表した図を示す。

下図の左は全原子座標の全データをメインプロセスから全プロセスにツリー型で送信し、右は send/recv での参照領域原子を隣接する cell にのみ渡すことになる。



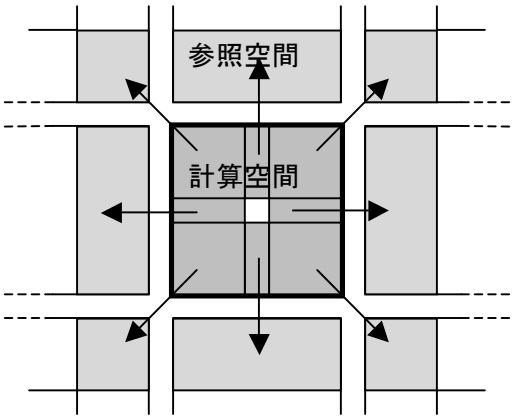
broadcast 型通信と send/recv 型通信

(2) 相互作用計算のための隣接 cell への原子情報の送受信

相互作用計算のための隣接 cell への通信は以下の２種類で、相補的な send/recv 通信で実行する。

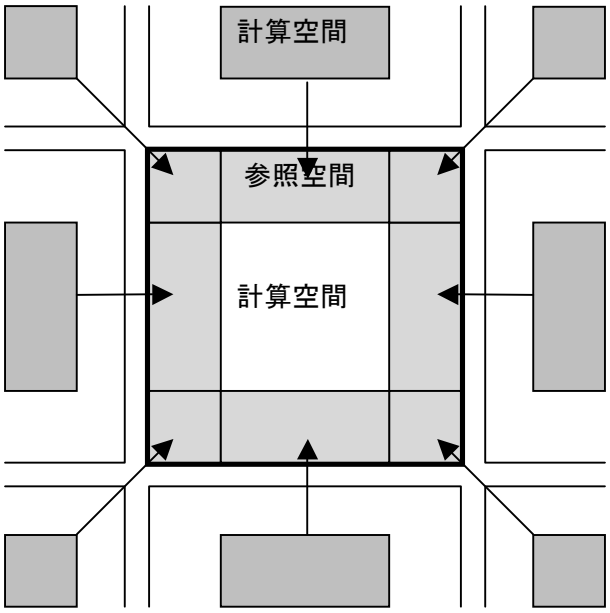
(2-1)当該 cell に含まれるオーバラップ領域の原子情報の隣接 cell の送信

当該 cell の計算空間に存在し、かつ、隣接 cell の参照空間に含まれる原子の ID と座標を送信する。



(2-2)隣接 cell に含まれるオーバラップ領域の原子情報の当該 cell への受信

当該 cell の参照空間に存在し、かつ、隣接 cell の計算空間に含まれる原子の ID と座標を受信する。



本試作での空間分割は 3 次元分割であり、1 回の系全体の原子情報の送受信では１つの cell に対して 26 回の原子 ID と座標の送信と 26 回の 受信が行われる。

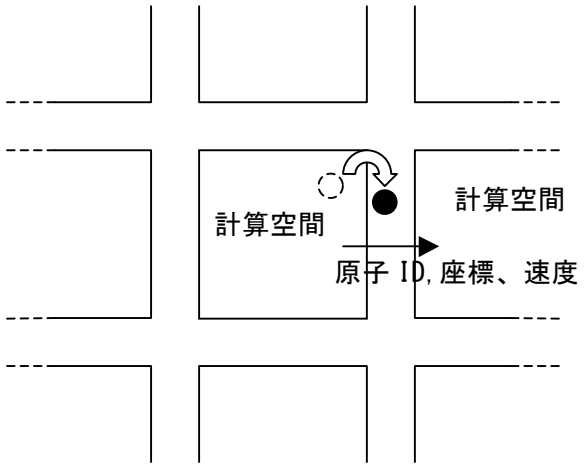
本通信は隣接していない cell 同士では順序性がないため、並列に送受信することは可能である。通信量はオーバラップ領域に含まれる原子数(≒オーバラップ領域の大きさ)に依存する。

(3)空間にまたがる原子の移動の通信

空間にまたがる原子情報の通信も相互作用計算のためと同様に、相補的な send/recv 通信で実行する。

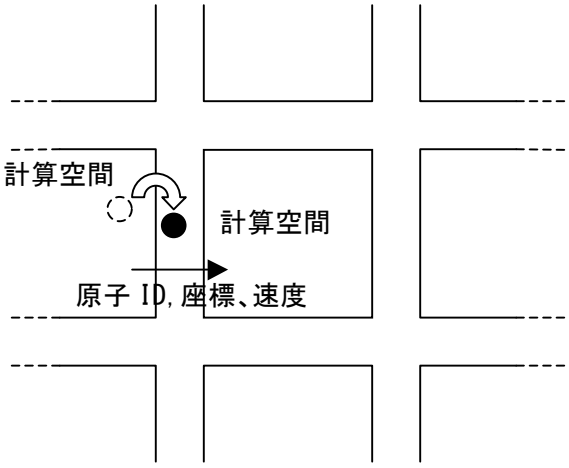
(3-1)当該 cell からの隣接セルに移動した原子情報の送信

当該 cell の計算空間外に移動した原子の ID、座標、速度を移動先の隣接 cell に送信する。



(3-2)隣接セルから当該 cell に移動した原子情報の受信

隣接 cell から当該 cell の計算空間内に移動した原子の ID、座標、速度を移動元の隣接 cell から受信する。



MD での⊿t 秒での原子移動は微少であり cell 間をまたがる移動を行う原子は少ない。

このことから、相互作用計算のための隣接 cell への原子情報の送受信に対し、本通信は通信量が少ないと想定される。

※本通信を cosgene の update と同様に、数ステップごとに実行することで通信オーバヘッドを削減するという効果が期待できる。

２．４．空間分割での相互作用計算・積分器の並列実行

空間分割では、相互作用計算による力、速度、座標更新は、cell 内の計算空間に所属する原子に対してのみ行う必要がある。また、これらの計算は cell 間の依存性はないため、基本的に並列実行可能である。

ただし、以下の手法を用いる場合は cell 間、または cell 全体での情報を統合する必要がある。

#	項目	計算内容	cell 情報の統合
1	温度制御	スケールファクター計算	運動エネルギーの総和
2	PME	逆空間項計算	メッシュの作成
3	系の回転・並進抑止	全体の	重心・運動量の総和
4	SHAKE	収束計算	収束判定値の通信
5	RIGID	剛体の力、トルク	隣接 cell 原子速度の通信
6	CAP	分子の重心計算	構成分子の座標の通信
7	分子ごとの温度制御	スケールファクター計算	分子ごとの運動エネルギーの総和
8	距離拘束	力の計算	対象原子の座標の通信

３．プロトタイプの仕様

"space.in"で指定された条件に従って系の空間分割を適用、mpirun で起動した各プロセスで cell 内原子の相互作用計算、積分器および cell 間のデータ通信を行い、1000 ステップ後の原子座標を出力するプロトタイプを作成した。

３．１．実行コマンド

本プロトタイプは下記のコマンドで起動する。

```
mpirun -np 並列台数 ./space
```

３．２．入力フォーマット

本プロトタイプはファイル"space.in"内に記載されたデータを読み込む。

1 行目：計算空間全体の下限の座標
2 行目：計算空間全体の上限の座標
3 行目：計算空間の各次元の分割数
4 行目：オーバラップ領域の距離 = カットオフ距離
5 行目：MD ループ回数
6 行目：原子数
7 行目以降：原子座標(PDB 形式)

```
-25.0  -25.0  -25.0      ; system space (bottom)
125.0  125.0  125.0      ; system space (top)
    4      4      4      ; division number
 12.0                    ; offset of space
 1000                    ; MD loop number
  676                    ; number of atoms
ATOM      1  Ar  ASP-    1      72.897  25.155  62.582 39.95  0.00
ATOM      2  Ar  ASP-    2      50.016  21.020  48.551 39.95  0.00
ATOM      3  Ar  ASP-    3      64.762  16.318  75.938 39.95  0.00
ATOM      4  Ar  ASP-    4      53.198  29.149  52.283 39.95  0.00
    :
```

３．３．出力

"fort."+(10+プロセス ID)のファイルに以下の情報を出力する。

(1)各プロセスの計算空間にある原子の初期座標

末尾が'STEP0"の行は、セル内に計算空間に配置された原子の ID, 座標を示す。

原子名はセルのインデックスの総和が奇数の場合は"O"、偶数の場合は"N"とする。

また、残基 ID には所属セルの ID を設定する。

添付の"select_1st_cord.sh"コマンドを実行することで、系全体の初期座標を表示することができる。

```
[kuro@waltz space]$ ./select_1st_cord.sh
ATOM      1  O   DUM    38      72.897  25.155  62.582 STEP0
ATOM      2  N   DUM    22      50.016  21.020  48.551 STEP0
ATOM      3  O   DUM    38      64.762  16.318  75.938 STEP0
ATOM      4  O   DUM    38      53.198  29.149  52.283 STEP0
ATOM      5  N   DUM    25      34.403  82.004  22.474 STEP0
ATOM      6  O   DUM    38      72.774  43.882  65.695 STEP0
ATOM      7  O   DUM    41      30.825  82.323  74.756 STEP0
ATOM      8  N   DUM    25      47.664  87.223  32.789 STEP0
ATOM      9  O   DUM    24       6.129  59.342  46.362 STEP0
```

(2)各プロセスの計算空間にある原子の最終座標

先頭が'FINAL"の行は､MD 終了時のセル内に計算空間に存在する原子の ID, 座標を示す。

添付の"select_2nd_cord.sh"コマンドを実行することで、MD 終了時の初期座標を表示することができる。

```
[kuro@waltz space]$ ./select_2nd_cord.sh
ATOM      1  O   DUM    38      72.807  25.151  62.649 FINAL
ATOM      2  N   DUM    22      50.036  20.969  48.487 FINAL
ATOM      3  O   DUM    38      64.611  16.490  75.807 FINAL
ATOM      4  O   DUM    38      53.160  29.117  52.220 FINAL
ATOM      5  N   DUM    25      34.415  81.856  22.628 FINAL
ATOM      6  O   DUM    38      72.752  43.812  65.731 FINAL
ATOM      7  O   DUM    41      30.972  82.047  74.535 FINAL
ATOM      8  N   DUM    25      47.664  87.400  32.711 FINAL
```

(3)各 cell での通信回数と通信バイト数、cell 間移動原子数

先頭が'#3"の行は、10,000 ステップでのセル毎の送受信回数、送受信バイト数、cell 間を移動した原子の個数を示す。

添付の"select_3rd_comm.sh"コマンドを実行することで、上記の情報を表示する。

```
[kuro@waltz space]$ ./select_3rd_comm.sh
fort.10:#3 CELL ID    =         0
fort.10:#3 SEND COUNT =    280000
fort.10:#3 SEND BYTES =  22400000
fort.10:#3 RECV COUNT =    140000
fort.10:#3 RECV BYTES =    560000
fort.10:#3 EXIST ATOM =        82
fort.10:#3 IN  COUNT  =         0
fort.10:#3 OUT COUNT  =         0
fort.10:#3 CELL ID    =         2
fort.10:#3 SEND COUNT =    200000
fort.10:#3 SEND BYTES =  10920000
fort.10:#3 RECV COUNT =    160000
fort.10:#3 RECV BYTES =   6440000
```

(4)ログ

標準出力に"space.in"ファイルで指定した系の情報、および MD 中のループ回数、エネルギー、1-5 相互作用数を表示する。

```
INFORMATION> SYSTEM
       SIZE  X= -25.0000000  125.0000000
             Y= -25.0000000  125.0000000
             Z= -25.0000000  125.0000000

       CUT OFF    =  12.0000000
       CELL NUMBER=     4     4     4
       CELL BOUND =  37.5000000   37.5000000   37.5000000
       MD LOOP       1000
       NUM OF ATOMS   676

 INFO>CUTOFF: 1-5 VDW + 1-5 HYD.BOND :        2906 +           0

 ********************************************************************************

 MD LOOP NUMBER        :               0 TIME (PSEC)           :         0.00000

 TOTAL      (KCAL/MOL) :  -0.5297778E+03 POTENTIAL  (KCAL/MOL) :  -0.5297779E+03
 KINETIC    (KCAL/MOL) :   0.1789969E-04
  VDW15  :  -0.5299659E+03  CAP.    :   0.1880511E+00

 INFO>CUTOFF: 1-5 VDW + 1-5 HYD.BOND :        2906 +           0

 ********************************************************************************

 MD LOOP NUMBER        :               1 TIME (PSEC)           :         0.00200

 TOTAL      (KCAL/MOL) :  -0.5297762E+03 POTENTIAL  (KCAL/MOL) :  -0.5297764E+03
 KINETIC    (KCAL/MOL) :   0.1617147E-03
  VDW15  :  -0.5299553E+03  CAP.    :   0.1788707E+00

 INFO>CUTOFF: 1-5 VDW + 1-5 HYD.BOND :        2906 +           0
```

３．４．制限事項

本試作では、以下の制限事項を設ける。

・系での力の計算は係数固定の電荷を持たない仮想的な原子の van der Waals 力と系外への原子の移動を抑止するパラメータ固定の CAP 拘束のみとする。
・積分器は、NVE の leap-frog のみとする。
・初期設定された空間外に原子が移動した場合の動作は保証しない。
・1 回の MD のループで、原子が２つ以上の cell をまたぐ移動を行う場合は動作を保証しない

３．５．インストールとテスト実行

(1)ファイルの展開

添付の space_decomposition.tar.gz を展開する。

展開後に以下のディレクトリが作成される。

・ソースディレクトリ："src/"
・テストディレクトリ："test/"
・比較用 cosgene 入力ファイル："compare/"

(2)ロードの作成

"src/"ディレクトリで"make"コマンドを実行するとロードモジュール"space"が作成される。

(3)テスト実行

"test/"ディレクトリに移動し、以下のコマンドを実行すると空間分割 MD プログラムが起動し、実行結果が標準出力に出力される。

./exec_1x1x1.sh	1cell での逐次実行
./exec_2x2x2.sh	8cell での 2 並列実行
./exec_4x4x4.sh	64cell での 2 並列実行

(4)同一条件の cosgene の実行

"compare/"ディレクトリに移動し、以下のコマンドを実行すると cosgene での比較用計算が実行される。

```
cosgene < md.inp
```

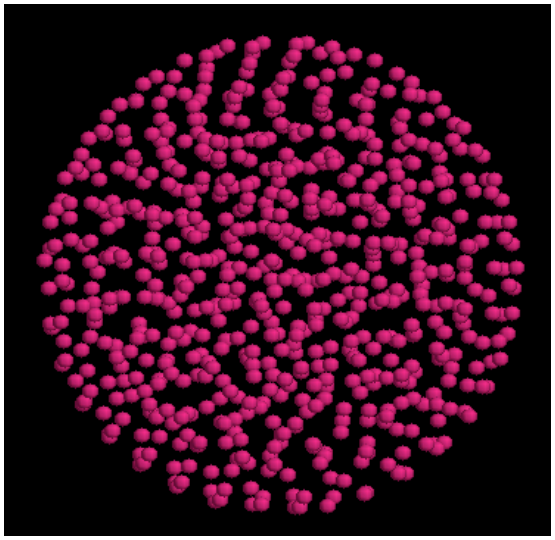
４．動作確認

仮想的な系に対し cosgene での NVE 計算を実行し総エネルギー、運動エネルギーの 1000 ステップでの比較を行った。

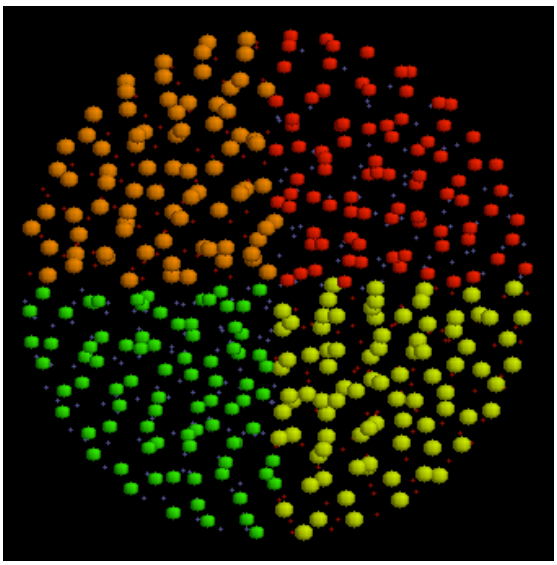
また、セル数、並列数での総エネルギー、運動エネルギーの 1000 ステップでの比較を行った。

(1)計算用の系と計算条件

比較用の系として 676 個の ARGON を模した系を作成した。



空間分割は 1*1*1, 2*2*2、4*4*4 の３つの分割パターンを設定した。



2*2*2 の空間分割

MD 条件として孤立系のカットオフ距離=12.0Å、⊿t = 2.0fs、中心=(50.0, 50.0, 50.0)、半径=45Åの CAP 拘束を適用した。

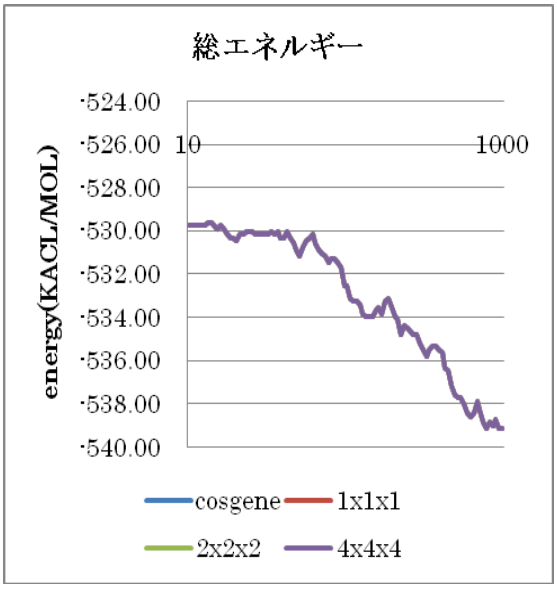
```
    EXE> MD
         LOOPLI=      1000
         LOGFOR=      DETA
         TIMEST=       2.0
         UPDATE=         1
         CUTMET=      ATOM        CUTLEN=     12.0D0
         DIEFUN=      CONS        DIEVAL=      1.0D0
         CALCAP=      CALC
         RADCAP=       45.0       ; FURCAP = 20.0D0
         FORCAP=      150.0       ; FUKCAP = 150.0D0
         FUNCAP=       HARMonic
```

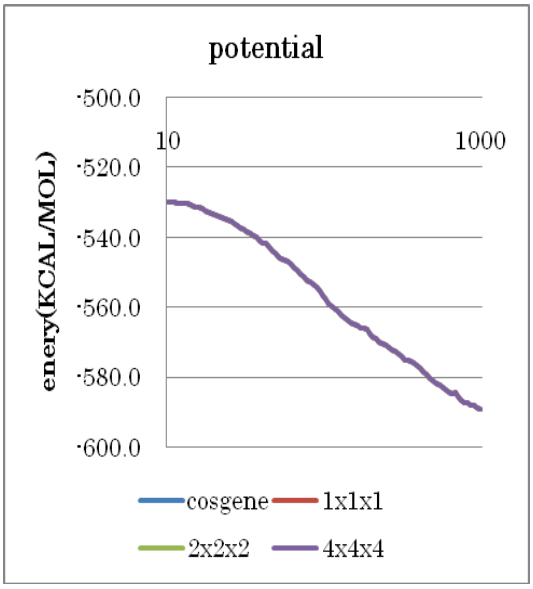
比較用 cosgene 実行の MD 制御指定

1000 ステップの実行結果で、全 MD 計算の最終総エネルギー、ポテンシャルエネルギーが小数点７ケタのレベルで一致したことを確認した。

1000 ステップ実行後の数値

	cell 数	並列数	終了時 total(KCAL/MOL)	終了時 potential(KCAL/MOL)
cosgene	-	-	-0.5391897E+03	-0.5891566E+03
空間分割 1x1x1	1	1	-0.5391897E+03	-0.5891566E+03
空間分割 2x2x2	8	2	-0.5391897E+03	-0.5891566E+03
空間分割 4x4x4	64	2	-0.5391897E+03	-0.5891566E+03





５．テーブル構成

(1)system_t

system テーブルは、系全体の空間情報、構成 cell 数を表現し、プロセスごとに一つ割り当てられる。

#	メンバ名	型、配列宣言	内容
1	Bottom	real*8,dimension(3)	系の座標下限
2	Top	real*8,dimension(3)	系の座標上限
3	Bound	real*8,dimension(3)	cell のサイズ
4	cutoffLength	real*8	カットオフ長=参照領域長
5	division	integer,dimension(3)	空間分割数
6	cellNum	integer	当該プロセスの cell 数-1
7	Rank	integer	プロセス ID
8	ProcessNum	integer	総プロセス数
9	ProcessID	integer,dimension(3)	セルに対応するプロセス ID
10	loopNum	integer	MD ループ回数

(2)cell_t

cell テーブルは、系を構成する cell を表現する。

cell の領域は、実行時に各プロセスに割り当てた個数分領域が確保されれる。

	要素名	型、配列宣言	内容
1	AtomNum	integer,dimension (-1:1,-1:1,-1:1)	計算領域/参照領域原子数 ＃(0,0,0)が計算領域
2	AtomIDList	integer,dimension (MAX_ATOM_IN_CELL, -1:1,-1:1,-1:1)	計算領域/参照領域の原子 ID のリスト
3	AtomCord	real*8,dimension (3,MAX_ATOM_IN_CELL, -1:1,-1:1,-1:1)	計算領域/参照領域の原子座標
4	AtomGrad	real*8,dimension (3,MAX_ATOM_IN_CELL, -1:1,-1:1,-1:1)	計算領域/参照領域の原子への力
5	AtomVel	real*8,dimension (3,MAX_ATOM_IN_CELL, -1:1,-1:1,-1:1)	計算領域/参照領域の原子速度
6	OverlapAtomNum	integer,dimension (-1:1,-1:1,-1:1)	オーバラップ領域原子数
7	OverlapAtomIDList	integer,dimension (MAX_ATOM_SUBCELL,	オーバラップ領域の原子 ID リスト

	要素名	型、配列宣言	内容
		-1:1,-1:1,-1:1)	
8	OverlapAtomCord	real*8,dimension (3, MAX_ATOM_SUBCELL, -1:1,-1:1,-1:1)	オーバラップ領域の原子 ID リストの原子座標
9	EnterAtomNum	integer,dimension (-1:1,-1:1,-1:1)	cell に追加される原子数
10	EnterAtomIDList	integer,dimension(MAX_ATOM_ENTER_CELL, -1:1,-1:1,-1:1)	cellに追加される原子 ID リスト
11	EnterAtomCord	real*8,dimension (3,MAX_ATOM_ENTER_CELL, -1:1,-1:1,-1:1)	cell に追加される原子の座標
12	EnterAtomVel	real*8,dimension (3,MAX_ATOM_ENTER_CELL, -1:1,-1:1,-1:1)	cell に追加される原子の速度
13	vdwCount	integer	cell 内 vdW 計算回数
14	VdwEnergy	real*8	cell 内 vdW energy
15	CapEnergy	real*8	cell 内 cap energy
16	KineticEnergy	real*8	cell 内運動エネルギー
17	bottom	real*8,dimension(3)	cell 範囲の下限座標
18	top	real*8,dimension(3)	cell 範囲の上限座標
19	bound	real*8,dimension(3)	cell の幅
20	recvRequest1	integer,dimension (-1:1,-1:1,-1:1)	cell 間通信の受信リクエスト(原子個数受信用)
21	recvRequest2	integer,dimension (-1:1,-1:1,-1:1)	cell 間通信の受信リクエスト(原子 ID 受信用)
22	recvRequest3	integer,dimension (-1:1,-1:1,-1:1)	cell 間通信の受信リクエスト(原子座標受信用)
23	recvRequest4	integer,dimension (-1:1,-1:1,-1:1)	cell 間通信の受信リクエスト(原子速度受信用)
24	sendCount	integer*8	当該 cell の総送信回数
25	sendBytes	integer*8	当該 cell の総送信データ量
26	recvCount	integer*8	当該 cell の総受信回数
27	recvBytes	integer*8	当該 cell の総受信データ量
28	outAtomCount	integer*8	cell から移動した原子数
29	inAtomCount	integer*8	cell に入ってきた原子数

６．モジュール一覧

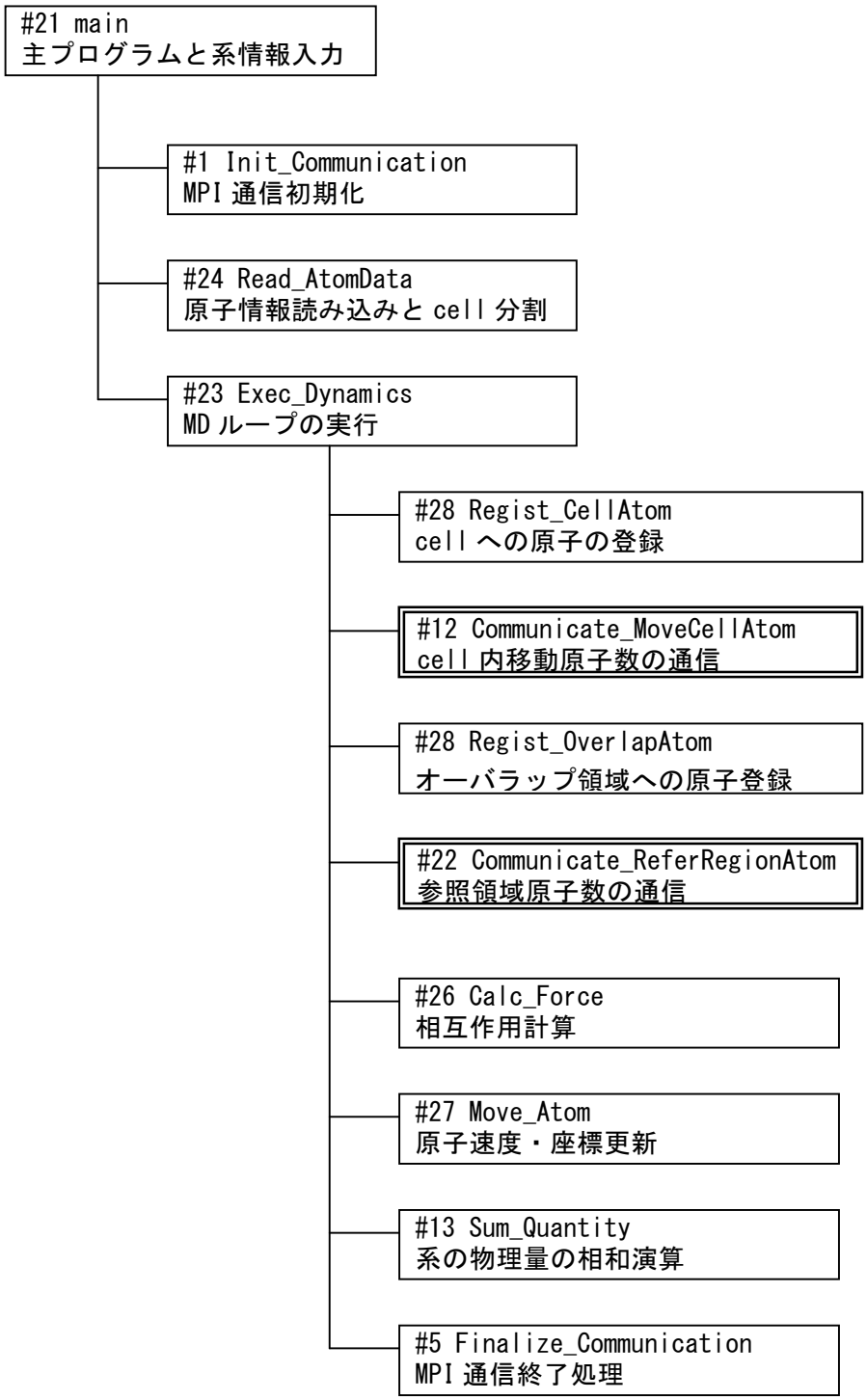
#	ファイル名	手続き名	機能
1	Space_Class.f90	Get_CellID	cell の x,y,z 位置を ID に変換
2		Get_CellIndex	対象 cell の ID を取得
3	Communicate_MoveAtom.f90	Init_Communication	MPI 通信初期化
4		Barrier	全プロセスのバリア同期
5		Finalize_Communication	MPI 通信終了
6		Send_MoveAtomNum	cell 外移動原子数の送信
7		Recv_MoveAtomNum	cell 内移動原子数の受信
8		Wait_MoveAtomNum	cell 内移動原子数の受信待ち
9		Send_MoveAtomCord	cell 外移動原子座標の送信
10		Recv_MoveAtomCord	cell 内移動原子座標の受信
11		Wait_MoveAtomCord	cell 内移動原子座標の受信待ち
12		Communicate_MoveCellAtom	cell 間移動原子通信の主制御
13		Sum_Quantity	倍精度スカラ値の総和
14	Communicate_ReferAtom.f90	Count_OverlapAtomNum	オーバラップ領域原子数取得
15		Get_OverlapAtomCord	オーバラップ領域原子座標取得
16		Send_ReferAtomNum	オーバラップ領域原子数送信(*1)
17		Recv_ReferAtomNum	参照領域原子数受信(*1)
18		Wait_ReferAtomNum	参照領域原子数受信待ち
19		Send_ReferAtomCord	オーバラップ領域原子座標送信
20		Recv_ReferAtomCord	参照領域原子座標受信
21		Wait_ReferAtomCord	参照領域原子座標受信待ち
22		Communicate_ReferRegionAtom	参照領域原子通信の主制御
23	Dynamics_Method.f90	Exec_Dynamics	MD ループの実行
24	Init_Atom.f90	Read_AtomData	原子情報読み込みと cell 分割
25	Init_Space.f90	-	主プログラムと系情報入力
26	Integrate_Method.f90	Calc_Force	相互作用計算
27		Move_Atom	原子速度・座標更新
28	Regist_Space.f90	Regist_CellAtom	cell への原子の登録
29		Regist_OverlapAtom	オーバラップ領域への原子登録

(1)当該 cell に属さずかつ参照する必要のある領域は参照領域、当該 cell に属しかつ他 cell から参照される領域をオーバラップ領域である。

実際には自 cell のオーバラップ領域から他 cell の参照領域へのデータ転送が行われる。

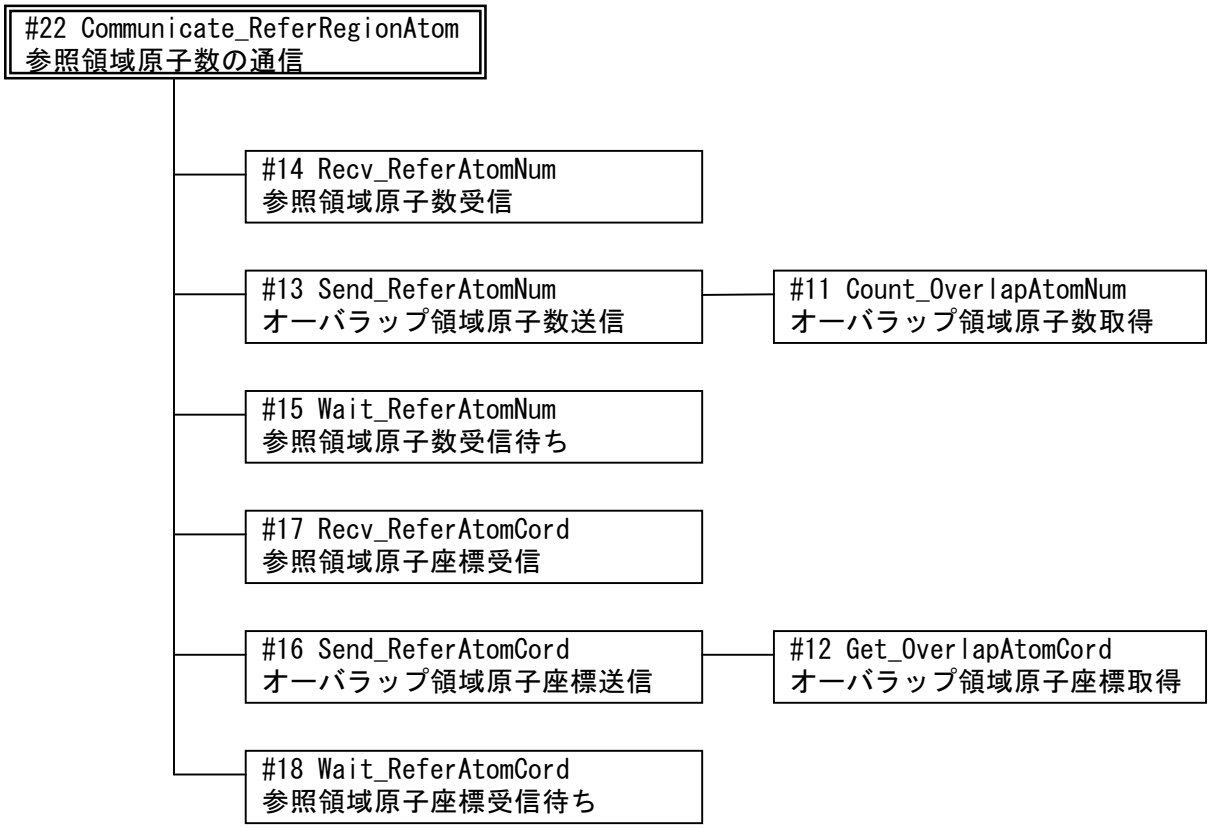
７．モジュール関連図

７．１．主制御



７．２．cell 間移動原子の通信

７．３．隣接 cell の参照領域原子の通信



８．モジュール一覧

８．１．Space_Class.f90

(1) function Get_CellID(ix,iy,iz,system) result (ID)

引数：

```
integer::ix
integer::iy
integer::iz
type(system_t)::system
```

戻り値：

```
integer::ID
```

機能：

cell の x,y,z 軸の３次元インデックスに対する１次元化した cell の ID を返す

```
function Get_CellIndex(cell, system) result (ID)
  type(system_t)::system
  type(cell_t)::cell
```

(2)function Get_CellIndex(cell, system) result (ID)

引数：

```
type(system_t)::system
type(cell_t)::cell
```

戻り値：

```
integer::ID
```

機能：

入力した cell の１次元化 ID を返す

８．２．Communicate_MoveAtom.f90

(1)subroutine Init_Communication(system)

引数：

```
type(system_t),intent(inout)::system          ! calculation space
```

戻り値：

なし

機能：

MPI 通信の初期化を行い、引数にプロセス数、プロセス ID を設定する。

(2)subroutine Barrier

引数：

なし

戻り値：

なし

機能：

全プロセスのバリア同期を実行する。

(3)subroutine Finalize_Communication

引数：

なし

戻り値：

なし

機能：

MPI 通信の終了処理を実行する。

(4)function Get_CommunicateTag(sx, sy, sz, rx, ry, rz, system) result (tag)

引数：

```
integer,intent(in)::sx  ! send cell x-axis
integer,intent(in)::sy  ! send cell y-axis
integer,intent(in)::sz  ! send cell z-axis
integer,intent(in)::rx  ! reveive cell x-axis
integer,intent(in)::ry  ! reveive cell y-axis
integer,intent(in)::rz  ! reveive cell z-axis
type(system_t),intent(in)::system          ! calculation space
```

戻り値：

```
integer::tag
```

機能：

送信側 cell のインデックスと受信側 cell のインデックスから通信タグを生成する。

(5)subroutine Send_MoveAtomNum(system, cell)

引数：

```
type(system_t),intent(in)::system          ! calculation space
type(cell_t),intent(inout)::cell           ! sub-space of current process
```

戻り値：

なし

機能：

当該 cell から移動する原子の個数を隣接 cell(26 個)に送信する。

(6)subroutine Recv_MoveAtomNum(system, cell)

引数：

```
type(system_t),intent(in)::system          ! calculation space
type(cell_t),intent(inout)::cell           ! sub-space of current process
```

戻り値：

なし

機能：

当該 cell に移動する原子の個数の非同期受信を発行する。

(7)subroutine Wait_MoveAtomNum(system, cell)

引数：

```
type(system_t),intent(in)::system          ! calculation space
type(cell_t),intent(inout)::cell           ! sub-space of current process
```

戻り値：

なし

機能：

当該 cell に移動する原子の個数の受信待ちを行う。

(8)subroutine Send_MoveAtomCord(system, cell)

引数：

```
type(system_t),intent(in)::system          ! calculation space
type(cell_t),intent(inout)::cell           ! sub-space of current process
```

戻り値：

なし

機能：

当該 cell から移動する原子の ID、座標、速度を隣接 cell(26 個)に送信する。

(9)subroutine Recv_MoveAtomCord(system, cell)

引数：

```
type(system_t),intent(in)::system          ! calculation space
type(cell_t),intent(inout)::cell           ! sub-space of current process
```

戻り値：

なし

機能：

当該 cell に移動する原子の ID、座標、速度の非同期受信を発行する。

(10)subroutine Wait_MoveAtomCord(system, cell)

引数：

```
type(system_t),intent(in)::system          ! calculation space
type(cell_t),intent(inout)::cell           ! sub-space of current process
```

戻り値：

なし

機能：

当該 cell に移動する原子の ID、座標、速度の受信待ちを行う。

(11)subroutine Communicate_MoveCellAtom(system, cells)

引数：

type(system_t),intent(inout)::system

type(cell_t),dimension(0:),intent(inout)::cellssubroutine

戻り値：

なし

機能：

当該 system に含まれる cell に対し、cell 間を移動する原子の原子数、ID、座標、速度の送受信を行う。

(12) subroutine Sum_Quantity(item)

引数：

real*8::item

戻り値：

なし

機能：

MPI_REDUCE 通信を使用し、入力データに対する全プロセスの総和の値を計算する。

８．３．Communicate_ReferAtom.f90

(1) function Count_OverlapAtomNum(cell, lx, ly, lz) result (num)

引数

```
type(cell_t),intent(in)::cell        ! sub-space of current process
integer,intent(in)::lx               ! x-direction of cell
integer,intent(in)::ly               ! y-direction of cell
integer,intent(in)::lz               ! z-direction of cell
```

戻り値：

```
integer::num     ! reference atom number of another cell
```

機能：

lx, ly, lz 方向の cell に対する参照領域(当該 cell のオーバラップ領域)の原子数を取得する。

(2)subroutine Get_OverlapAtomCord(cell, num, cord, id, ix, iy, iz)

引数

```
type(cell_t),intent(in)::cell        ! sub-space of current process
integer,intent(inout)::num
real*8,dimension(3,MAX_ATOM_ENTER_CELL)::cord
integer,dimension(MAX_ATOM_ENTER_CELL)::id
integer,intent(in)::ix               ! relative x index of neighbour cell
integer,intent(in)::iy               ! relative y index of neighbour cell
integer,intent(in)::iz               ! relative z index of neighbour cell戻り
```

戻り値：

なし

機能：

lx, ly, lz 方向の cell に対する参照領域(当該 cell のオーバラップ領域)の原子数、原子 ID、座標を引数の num, cord, id に設定する。

(3)subroutine Send_ReferAtomNum(system, cell)

引数：

```
type(system_t),intent(in)::system          ! calculation space
type(cell_t),intent(inout)::cell           ! sub-space of current process
```

戻り値：

なし

機能：

当該 cell のオーバラップ領域の原子の個数を隣接 cell(26 個)に送信する。

(4)subroutine Recv_ReferAtomNum(system, cell)

引数：

```
type(system_t),intent(in)::system          ! calculation space
type(cell_t),intent(inout)::cell           ! sub-space of current process
```

戻り値：

なし

機能：

当該 cell の参照領域の原子数の非同期受信を発行する。

(5)subroutine Wait_ReferAtomNum(system, cell)

引数：

```
type(system_t),intent(in)::system          ! calculation space
type(cell_t),intent(inout)::cell           ! sub-space of current process
```

戻り値：

なし

機能：

当該 cell の参照領域の原子数の受信待ちを行う。

(6)subroutine Send_ReferAtomCord(system, cell)

引数：

```
type(system_t),intent(in)::system          ! calculation space
type(cell_t),intent(inout)::cell           ! sub-space of current process
```

戻り値：

なし

機能：

当該 cell のオーバラップ領域の原子の ID、座標を隣接 cell(26 個)に送信する。

(7)subroutine Recv_ReferAtomCord(system, cell)

引数：

```
type(system_t),intent(in)::system          ! calculation space
type(cell_t),intent(inout)::cell           ! sub-space of current process
```

戻り値：

なし

機能：

当該 cell の参照領域の原子 ID、座標の非同期受信を発行する。

(10)subroutine Wait_ReferAtomCord(system, cell)

引数：

```
type(system_t),intent(in)::system          ! calculation space
type(cell_t),intent(inout)::cell           ! sub-space of current process
```

戻り値：

なし

機能：

当該 cell の参照領域の原子 ID、座標の受信待ちを行う。

８．４．Dynamics_Method.f90

(1)subroutine Exec_Dynamics(system, globalSpace, cells)

引数：

```
type(system_t)::system
type(globalSpace_t)::globalSpace
type(cell_t),dimension(0:)::cells
```

戻り値：

なし

機能：

当該プロセスに含まれるcellのMDループを実行する。

８．５．Init_Atom.f90

(1)subroutine Read_AtomData(system, globalSpace, cells, atomNum)

引数：

```
type(system_t)::system
type(globalSpace_t)::globalSpace
type(cell_t),dimension(0:)::cells
integer::atomNum
```

戻り値：

なし

機能：

当該プロセスに含まれるcellに対し、space.inから入力した原子を割り当てる。

８．６．Integrate_Method.f90

(1)subroutine Calc_Force(cell, system)

引数：

```
type(cell_t)::cell
type(system_t)::system
```

戻り値：

なし

機能：

当該 cell に対し、所属する原子の CAP 拘束と van der Waals 力の計算を行い、cell 単位のポテンシャルエネルギーと各原子への力を計算する。

(2)subroutine Move_Atom(cell, system, dt, firstCoeff)

引数：

```
type(cell_t)::cell
type(system_t)::system
real*8::firstCoeff      ! initial leap frog = 0.5, other = 1.0
real*8::dt
```

戻り値：

なし

機能：

当該 cell に対し、所属する原子の速度と座標を更新する。

また、当該時刻での当該 cell 内原子の運動エネルギーを計算する。

８．７．Regist_Space.f90

(1)subroutine Regist_CellAtom(cell, system)

引数：

type(cell_t)::cell

type(system_t)::system

戻り値：

なし

機能：

当該 cell の所属原子に対し、現在の座標に対する所属 cell(当該 cell または隣接 cell)を設定する。

(2)subroutine Regist_OverlapAtom(cell, system)

引数：

type(cell_t)::cell

type(system_t)::system

戻り値：

なし

機能：

当該 cell の所属原子に対し、所属するオーバラップ領域を設定する。
