Vol.2014-ARC-213 No.24
Vol.2014-HPC-147 No.24
2014/12/10

GPUによる近接相互作用に基づく粒子計算の近傍探索手法

渡辺 勢也[1,a)]　青木 尊之[1,b)]　都築 怜理[1,c)]　下川辺 隆史[1,d)]

概要：近接相互作用に基づく粒子法の一つである個別要素法の GPU での計算に，近傍粒子探索手法である粒子登録法とセル分割法（リンクリストを用いた方法とハッシュを用いた方法）を実装し，各手法の違いによる計算時間およびメモリ使用量の比較を行う．さらに，粒子登録法とリンクリストを用いたセル分割法のハイブリット手法および粒子登録法とハッシュを用いたハイブリット手法を実装し高速化を行った．近傍探索手法を比較する例題として，100 万個の粒子を用いた 3 次元個別要素法によるダム崩壊問題をそれぞれの近傍探索手法で行った．粒子登録法とリンクリストを用いたセル分割法のハイブリット手法が実装した 5 つの近傍探索手法のなかで最も計算を高速に行えることがわかった．これは，周囲のセルに含まれる粒子を参照するのがハッシュ法よりも速いリンクリストを用いたことと，粒子登録法により相互作用計算で影響半径外の粒子との無駄な計算を極力減らしたためである．メモリ使用量はハッシュ法が最も少なく，使用できるメモリ量が限られている GPU の場合はハッシュ法を用いればよいことがわかった．また，粒子の物理量のデータ構造による計算時間への影響について検討を行い，粒子データを Structure Of Array とすると，Array Of Structure とした場合と比べ，計算時間を短縮できることを確認した．

1. 緒言

近接相互作用に基づく粒子法には，流体を計算するための MPS(Moving Particle Semi-Implicit) 法 [1] や SPH(Smoothed Particle Hydrodynamics) 法 [2] や，粉体を計算するための個別要素法である DEM(Distinct Element Method)[3] などがある．実際の現象を粒子法で扱うには多くの粒子を用いた大規模な計算が必要であり，現実的な時間で計算を終わらせるためには，粒子法の高速化が必要不可欠である．

近接相互作用に基づく粒子法の高速化の手段として，近傍粒子探索の効率化があげられる．近接相互作用に基づく粒子法では，粒子からある一定距離内に含まれる近傍の粒子との相互作用を毎ステップ計算する．全粒子に対して近傍の粒子であるかの判定を行った場合，計算コストは全粒子数の 2 乗のオーダーとなる．近傍粒子の探索を効率化する手法として，粒子登録法 [4] やセル分割法がある．粒子登録法は，各粒子が自身の近傍粒子のインデックスを記憶しておき，ある一定ステップの間はリスト内の粒子とのみ相互作用計算を行う手法である．セル分割法は，計算領域を一様な格子に分割し，近傍探索のときは周囲のセル内に含まれる粒子に対してのみ，相互作用の計算を行う方法である．このような近傍粒子探索の高速化を行ったとしても，所望の計算を行うための計算がかかりすぎるために CPU の 1core で計算できる粒子数は数十万個程度である．

粒子法の計算のさらなる高速化および大規模化を行うために，最近では画像処理に特化した演算器である GPU(Graphics Processing Unit) を利用した，粒子法の研究が盛んに行われている [5], [6], [7]．参考文献 [8] は，複数 GPU で大規模粒子計算を可能にするための動的負荷分散手法を提案している．参考文献 [9] と [10] は，複数の GPU を利用して SPH 法や DEM の大規模計算を行っている．

複数 GPU を用いた粒子計算では，1 台の GPU にできるだけ多くの粒子を割り当てることで，さらに大規模な計算が実現できる．そのためには，メモリ使用量を抑えた粒子法の高速化手法を実装する必要がある．本研究の目的は，GPU 計算における近接相互作用に基づく粒子計算の最適な近傍探索手法を定めることである．本論文では，3 次元 DEM の GPU での計算に，粒子登録法と，リンクリストを用いたセル分割法 [11] と，ハッシュを用いたセル分割法 [12] と，粒子登録法とリンクリストを用いたセル分割法のハイブリット手法と，粒子登録法とハッシュを用いたセル分割法のハイブリット手法の 5 つ近傍探索手法を実装し，計算時間およびメモリ使用量の比較を行う．また，粒子の物理量のデータ構造を Array Of Structure と Structure Of

[1] 東京工業大学
Tokyo Institute of Technology, Meguro, Tokyo 152-8550, Japan
a) watanabe@sim.gsic.titech.ac.jp
b) taoki@gsic.titech.ac.jp
c) tsuzuki@sim.gsic.titech.ac.jp
d) shimokawabe@sim.gsic.titech.ac.jp

Array とした場合の計算時間の比較を行う．

本論文の以降で示す計算の実行環境は，東京工業大学学術国際情報センターの TSUBAME2.5 を用いる．使用した GPU は NVIDIA 社の Tesla K20X であり，CUDA の Version は 6.0 である．

2. DEM(Distinct Element Method)

DEM は粉体群を構成している個々の粒子の運動を解くことにより全体の挙動を表現する計算手法である．重力などの外力と粒子間力の合力を計算して粒子に働く力を求め，時間積分して粒子の位置と速度を更新していく．粒子間の相互作用は，**図 1** に示すようにバネとダッシュポットにより表現され，粒子同士が接触している場合にのみ働く．バネによる力は粒子同士の食い込み深さに比例し，ダッシュポットは粒子間の相対速度に比例した減速力を粒子に与える．粒子に加わる力は法線方向に働く反発力と，接線方向に働く摩擦力に分解し，接線方向のスライダーは粒子の摩擦係数に応じて接線方向の力の上限を与える．粒子 i が粒子 j から受ける法線方向および接線方向の力 $\mathbf{F}_{ij}^N$，$\mathbf{F}_{ij}^T$ は以下の式によって表される．

$$\mathbf{F}_{ij}^N = k^N \mathbf{L}_{ij}^N + c^N \frac{\Delta \mathbf{L}_{ij}^N}{\Delta t} \tag{1}$$

$$\mathbf{F}_{ij}^T = k^T \mathbf{L}_{ij}^T + c^T \frac{\Delta \mathbf{L}_{ij}^T}{\Delta t} \tag{2}$$

ここで，$\mathbf{L}$ はバネの圧縮量ベクトル，$\Delta\mathbf{L}$ は相対変位増分ベクトル，Δt は時間刻み，k はバネの弾性係数，c はダッシュポットにおける粘性減衰係数である．添字 N と T はそれぞれ法線方向と接線方向を表す．粒子半径を R_i とすると，接線方向の力による粒子 i に加わるモーメントのベクトル $\mathbf{M}_{ij}$ は次のように表される．

$$\mathbf{M}_{ij} = \frac{R_i}{|\mathbf{x}_j - \mathbf{x}_i|}(\mathbf{x}_j - \mathbf{x}_i) \times \mathbf{F}_{ij}^T \tag{3}$$

図 2 のように粉体の一つの粒子は一度に複数個の周囲の粒子と衝突する可能性があり，粒子 i と接触しているすべての粒子 j に関して力とモーメントベクトルを計算し，粒子 i に作用する力とモーメントの合力ベクトルを求める．粒子の並進および重心を中心とする回転の運動方程式は以下で式で表される．

$$m_i \frac{d^2\mathbf{x}_i}{dt^2} = \sum_{i \neq j} [\mathbf{F}_{ij}] + m_i \mathbf{g} \tag{4}$$

$$I_i \frac{d^2\theta_i}{dt^2} = \sum_{i \neq j} [\mathbf{M}_{ij}] \tag{5}$$

ここで，$\mathbf{x}_i$ と θ_i は粒子 i の位置および回転角ベクトル，m_i と I_i は粒子の質量および慣性モーメントである．

得られた各粒子に関する並進および回転の運動方程式を数値積分することで，新しい時刻における粒子の位置，速度，回転角および角速度を求める．

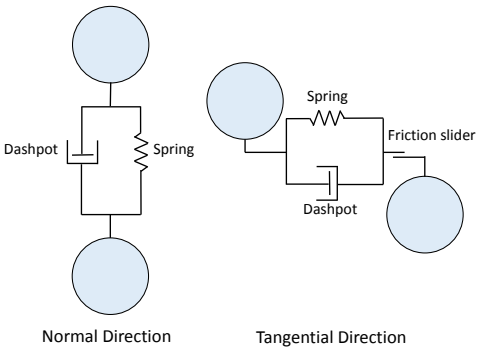

図 1 個別要素法の粒子間相互作用モデル
Fig. 1 DEM interaction model.

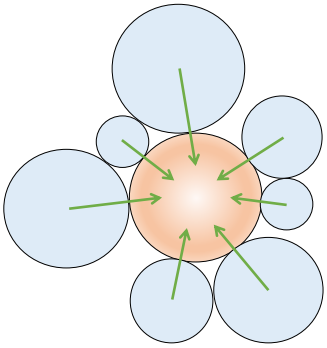

図 2 近傍粒子との衝突
Fig. 2 Collisions among contacting particles.

3. セル分割法による近傍探索手法

3.1 セル分割法

DEM などの近接相互作用に基づく粒子法では，ある一定距離内に含まれる近傍粒子との相互作用を毎ステップ計算する．近傍粒子を探索する際に，系を構成している N 個のすべての粒子との距離を計算した場合，ひとつの粒子に対して N-1 回の他の粒子との距離計算が必要となり，計算のオーダーは N^2 に比例する．そのため，粒子数を大きくすると，近傍粒子の探索に計算時間のほとんどが費やされてしまう．DEM では，粒子間の相互作用は接触している粒子との間にのみ発生するので，接触していない粒子との距離計算および相互作用計算は行う必要がない．そのため，接触している粒子を効率よく探索すれば，近傍粒子探索にかかる時間を大幅に削減でき，計算時間の短縮が可能である．

図 3 の左図のように計算領域を一様な格子で分割し，各粒子がどのセルに含まれるかを調べる．近傍粒子探索の際は，注目している粒子の所属しているセルとその周囲のセルに含まれる粒子とのみ距離計算を行う．セルの大きさは，影響半径よりも大きくすると，ひとつのセルに含まれる粒子が増え，近傍粒子探索にかかる時間が増える．

DEM の場合は，セルの大きさは最大粒子直径と等しく

Vol.2014-ARC-213 No.24
Vol.2014-HPC-147 No.24
2014/12/10

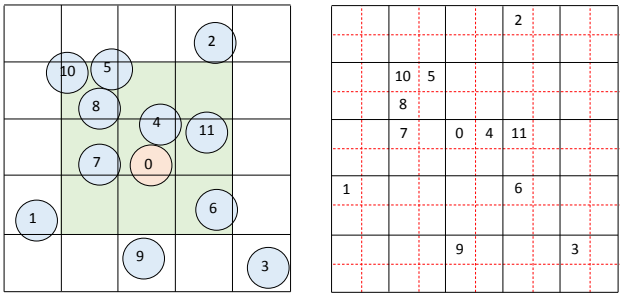

図 3　セル分割による近傍探索
Fig. 3　Neighboring cell list.

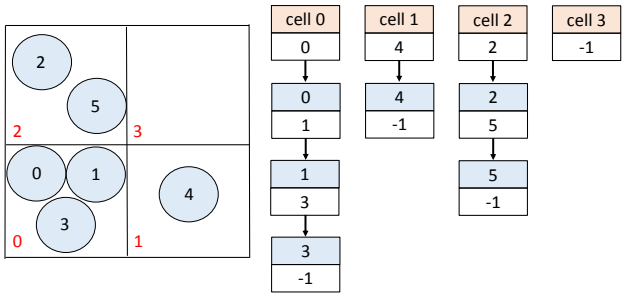

図 4　リンクリストを用いたセル分割法による近傍探索
Fig. 4　Neighboring cell list using linked-list method.

設定する．全粒子の大きさが同じの場合，ひとつのセルに含まれる最大の粒子数は 2 次元計算で 4 個，3 次元計算で 8 個である．図 3 の右図のように各セルはセル内に粒子が存在していない場合でも，粒子インデックスを格納するメモリ領域を持っておく必要がある．そのため，セル分割法を行うためには，2 次元計算で格子数の 4 倍，3 次元計算で 8 倍のデータ数が必要となり，計算領域を広く取るとメモリが枯渇する可能性がある．大きさの異なる粒子を扱う場合は，ひとつのセルに対してさらに多くの粒子が入る可能性があり，各セルが確保しておくメモリ量が多くなる．

セル分割法でのメモリ使用量を抑える方法として，3.2 節，3.3 節で示す，リンクリストを用いる方法とハッシュを用いる方法がある．

3.2　リンクリストを用いたセル分割法

図 4 にリンクリストを用いたセル分割法（以下，リンクリスト法と記す）による近傍探索手法の概念図を示す．各セルはセル内の粒子のインデックスをすべて持つのではなく，先頭の粒子インデックスのみを格納する．セル内に粒子が存在しない場合は，粒子が存在しないことを表す -1 を格納する．各粒子は同一セル内の他の粒子のインデックスをひとつ記憶する．最後尾の粒子には，リストが終了することを表す -1 を記憶させる．近傍粒子探索の際は，周囲のセルのリストを辿って行くことにより，近傍粒子のインデックスを参照する．

近傍探索に必要なデータは，各セルのもつ先頭粒子のインデックスと，各粒子のもつ次の粒子のインデックスである．そのため，計算領域を広げて空間の分割数が増えたときに，セル分割法ほどメモリ使用量が大きく増えない．

リンクリストの作成では，粒子同士のリンクの作成する際に逐次処理が必要となる．GPU 上でリンクリストを作成する際は，CUDA により提供されている atomic 関数の atomicExch を利用する．

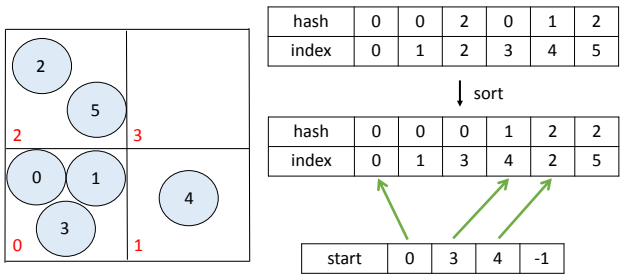

図 5　ハッシュを用いたセル分割法による近傍探索
Fig. 5　Neighboring cell list using hash method.

3.3　ハッシュを用いたセル分割法

図 5 にハッシュを用いたセル分割法（以下，ハッシュ法と記す）の概念図を示す．各粒子の所属するセルの番号を hash 配列に格納し，その番号を用いて粒子のインデックスを登録してある index 配列のソートを行う．GPU 上でのソートには，Thrust ライブラリの sort_by_key を使用する．次に，同じセル番号を持つ粒子が連続する部分の始まりの位置を start 配列に記憶させる．セル内に粒子が含まれていない場合は，粒子が存在しないことを表す -1 を記憶させる．近傍探索の際は，周囲のセルの番号とセル内粒子の始点を取得して，所属しているセルの番号が変わるまで粒子のインデックスを読み込む．

4.　粒子登録法による近傍探索手法

粒子登録法は，近傍粒子のインデックスを各粒子が近傍リストとして記憶し，ある一定時間中は近傍リスト内の粒子とのみ相互作用計算を行う．**図 6** に粒子登録法の概念図を示す．注目している粒子 i に対して，影響半径 rc よりも大きい半径 Rc を設定し，Rc 内に含まれる粒子を近傍リストに格納する．Rc の大きさの設定は，

$$Rc = rc + \alpha D_{min} \tag{6}$$

で設定する．ここで，D_{min} は最小の粒子直径，α は近傍リストの更新頻度を調整するパラメータである．近傍リス

Vol.2014-ARC-213 No.24
Vol.2014-HPC-147 No.24
2014/12/10

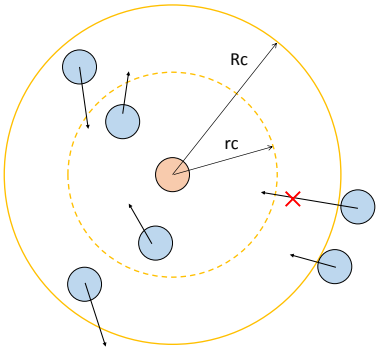

図 6　粒子登録法による近傍探索
Fig. 6　Book-keeping method.

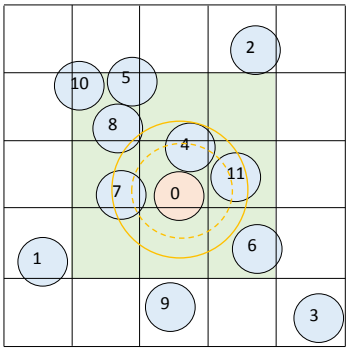

図 7　粒子登録法とセル分割法のハイブリットによる近傍探索
Fig. 7　Combination method of Neighboring cell list with book-keeping method.

トに含まれる粒子とのみ相互作用を計算することで，遠方の粒子との無駄な相互作用計算が必要無くなり，近傍探索にかかる時間を削減できる．近傍リストは 1 ステップ毎に更新するのではなく，図 6 の赤いバツ印がついた粒子のように，近傍リストに含まれていない粒子が注目している粒子の影響半径 rc 内に入った場合に更新を行う．実際の実装では，各ステップで全粒子の中で最大の速度を求めて，その移動距離を粒子移動距離 $xbook$ に積載していく．

$$xbook+ = v_{max}\Delta t \tag{7}$$

$xbook$ が $(Rc - rc)/2$ を上回った場合，全粒子の近傍リストの更新を行う．このとき，粒子移動距離の積載値 $xbook$ を初期化する．

粒子登録法では，近傍リストに入る最大の粒子数の分だけ各粒子が配列を確保しておく必要がある．そのため，必要なメモリ量は粒子数に比例して大きくなる．セル分割法と異なり，必要なメモリ量が計算領域に比例して増えない．

5. 粒子登録法とセル分割による近傍探索手法のハイブリット手法

図 7 に粒子登録法とセル分割による近傍探索手法を組み合わせた手法の概念図を示す．粒子登録法の近傍リストの更新では，全粒子との距離の計算が必要となり，近傍リストの作成に粒子数の 2 乗の計算コストがかかる．近傍リストの作成のときに，セル分割法による近傍探索を用いることで，近傍リストの作成を高速化することが可能となる．空間を分割するセルの大きさは，影響半径の大きさではなく，近傍リストを登録する半径 Rc とする．これにより，粒子が所属するセルと周囲のセルに含まれる粒子との距離を計算するだけで，リストの作成が可能となる．図 7 のように粒子登録法により，周囲セルに含まれる粒子からさらに近傍の粒子を絞り込むため，セル分割による近傍探索に比べて相互作用計算で参照する粒子数が少なくなり，相互作用計算の高速化が可能である．

本研究では，粒子登録法とリンクリスト法を組み合わせたハイブリット手法と，粒子登録法とハッシュ法のハイブリット手法を実装する．

6. 粒子のデータ構造による計算時間への影響

GPU での粒子法の計算では，メモリアクセスが計算時間を律速する．そこで，粒子の物理量のデータ構造を変えることによる計算時間への影響を調べる．粒子のデータ構造を Array Of Structure(AOS) および Structure Of Array(SOA) として計算時間の比較を行った．AOS は，ひとつ粒子が持つすべての物理量をグループとし，隣接したメモリに格する．SOA は，各物理量の配列を隣接したメモリに格納する．

速度比較を行う例題として，3 次元 DEM による 4 万個の粒子を用いたダム崩壊問題を扱う．計算条件を**表 1** に示す．粒子の物理量はすべて倍精度であり，数値積分には 2 段 2 次のルンゲクッタ法を用いる．近傍粒子探索の手法は，粒子登録法とリンクリスト法のハイブリット手法，および粒子登録法とハッシュ法のハイブリット手法とする．

粒子のデータ構造を AOS と SOA としたときの計算時間を**図 8** に示す．緑色が粒子間相互作用の計算に要した時間を，青色が粒子登録法の近傍粒子リストの作成にかかった時間を，黄色が 2 段 2 次ルンゲクッタ法による時間発展の計算時間を示す．

粒子のデータ構造を SOA とした場合は，AOS とした場合と比べ，2 つの手法の平均で，粒子間相互作用は 91%，近傍粒子リストの作成は 62%，時間発展は 15%，全計算時間は 73%に短縮されている．時間発展のときに必要な物理量は，粒子の位置，速度，加速度，質量などであるが，データ構造が AOS の場合は，接線方向のバネの圧縮量など時間発展に用いられない物理量を保持しているメモリにアクセスしてしまう．データ構造を SOA とすると，時間発展に必要な物理量のメモリのみを参照するため，特に，時間発展の部分が大幅に短縮されたと考えられる．

粒子のデータ構造を SOA とすると，AOS とした場合よりも計算時間を短縮できることが確認できたので，7 章の近傍探索手法の比較では，粒子のデータ構造は SOA とする．

Vol.2014-ARC-213 No.24
Vol.2014-HPC-147 No.24
2014/12/10

表 1　計算条件
Table 1　Physical condition.

Number of particle	40,000
Size of calculating area	$2.0 \times 1.0 \times 2.5$ [m^3]
Particle diameter	1.25×10^{-2} [m]
Particle mass	0.04 [kg]
Spring constant (Normal)	5.0×10^{5} [N/m]
Spring constant (Tangential)	5.0×10^{4} [N/m]
Damping coefficient (Normal)	2.0×10^{1} [Ns/m]
Damping coefficient (Tangential)	2.0 [Ns/m]
Coefficient of friction	0.3
Discrete time	1.0×10^{-5} [sec]

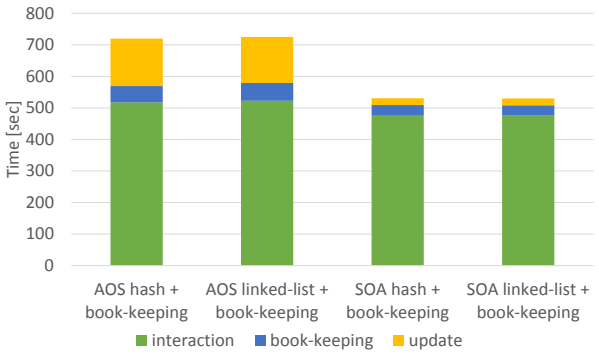

図 8　粒子物理量のデータ構造による計算時間への影響
Fig. 8　Comparison between Array of Structure and Structure of Array.

7.　近傍探索手法の性能比較

7.1　性能比較に用いる問題の設定

3 章，4 章および 5 章で示した近傍探索手法を実装した DEM 計算のプログラムで，100 万個の粒子を用いたダム崩壊シミュレーションを行い，計算時間を比較する．詳しい計算条件は**表 2** に示す．

図 9 の左側の図に示すようにダムの初期配置を作成する．粒子の色は，粒子の速さが大きいほど赤く，小さいほど青くしている．この初期配置は，粒子を格子点状に均等に並べたあと，そこから擬似乱数を用いてわずかに各粒子の配置をずらして配置し，自由落下させ安定になるまで待ったものである．図 9 の右側の図は，初期状態から 60,000 ステップの計算を行ったときの状態ある．図 9 の左図の状態では粒子がほぼ静止しており，粒子登録法の近傍リストの更新が殆ど行われないため，粒子がある程度の速度で移動している右図の状態を，初期配置として近傍粒子探索手法の性能比較を行う．5,000 ステップの計算を行い，各近傍探索手法の実行時間を計測する．粒子の物理量は倍精度であり，時間発展には 2 段 2 次のルンゲクッタ法を用いる．

7.2　リンクリスト法とハッシュ法の比較

近傍探索手法をリンクリスト法およびハッシュ法とした場合の計算時間を**表 3** に示す．左から，粒子間相互作用の計算，リンクリストの作成，ハッシュ法，2 段 2 次ルンゲクッタ法による時間発展，全体の計算にかかった時間である．

リンクリストの作成にかかる時間は 23.55 sec であるのに対して，ハッシュ法は 41.89 sec であり，リンクリスト法はハッシュ法よりも 1.75 倍高速であることがわかる．しかし，リンクリストの作成にかかる時間は，全体の計算のうちの 1.2%で，ハッシュ法は全体の 2.0%であるため，全体の計算時間に与える影響は少ない．相互作用計算にかかる時間は，リンクリスト法を用いた場合のほうがハッシュ法よりも短い．相互作用計算におけるリンクリスト法とハッシュ法の違いは，周囲のセルに含まれる粒子のインデックスを取得するところだけであるので，リンクリスト法のほうが近傍粒子探索が高速である．

リンクリストの作成やハッシュ法での粒子インデクスのソートにかかる時間は，全体の計算時間に対して小さく，粒子間相互作用の計算が計算時間の殆どを占めていることが確認できる．計算を高速化するためには，粒子間相互作用の計算を高速化することが必要である．

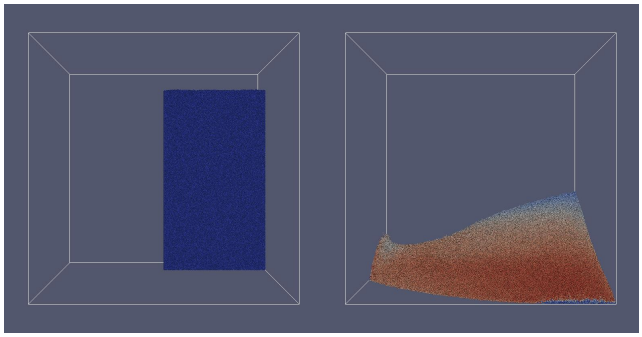

図 9　ダム崩壊問題初期配置（左）と性能評価に用いる初期配置（右）
Fig. 9　Initial condition of braking dam problem for paformance test.

表 2　計算条件
Table 2　Physical condition.

Number of particle	1,000,000
Size of calculating area	$0.4 \times 0.4 \times 0.4$ [m^3]
Particle diameter	1.0×10^{-3} [m]
Particle mass	1.0×10^{-5} [kg]
Spring constant (Normal)	4.0×10^{3} [N/m]
Spring constant (Tangential)	1.6×10^{3} [N/m]
Damping coefficient (Normal)	0.4 [Ns/m]
Damping coefficient (Tangential)	0.25 [Ns/m]
Coefficient of friction	0.3
Discrete time	5.0×10^{-6} [sec]

7.3　粒子登録法とリンクリスト法とハッシュ法の比較

まず，粒子登録法で近傍粒子リストを作成する半径を決

表 3　リンクリスト法とハッシュ法の計算時間
Table 3　Calculation time of linked-list method and hash method.

	interaction	linked-list	hash	update	total
linked-list	1930.21	23.55	—	19.50	1973.74
hash	1998.21	—	41.89	18.77	2058.93

unit : sec

定するパラメータ α を変化させ，近傍粒子リストの更新頻度を変えた場合の計算時間を調べる．パラメータ α と近傍粒子リストを作成する半径の関係は式 (6) である．パラメータ α を 0.2 から 2.0 まで変化させたときの計算時間を**図 10** に示す．緑色が粒子間相互作用の計算に要した時間を，青色が粒子登録法の近傍粒子リストの作成に要した時間である．

パラメータ α が小さいほどリスト内の粒子の数が少なく相互作用計算が高速に行えるが，近傍粒子リストの更新頻度が短くなるためリスト作成の回数が増え，リスト作成にかかる時間が増える．一方，パラメータ α を大きくすると，リスト内に接触していない粒子が増え，相互作用計算時に無駄な計算が多くなり時間がかかるが，リストの更新頻度が少ないため，リスト作成にかかる時間は短くなる．近傍粒子リストの作成時には，他のすべての粒子との距離計算が必要であるため，リスト作成の計算時間は N^2 に比例する．そのため，近傍粒子リストの半径を大きく取り，リスト更新頻度を少なくするほうが，全体の計算時間の短縮になる．

次に，表 3 のリンクリスト法とハッシュ法による近傍粒子探索と粒子登録法の計算時間の比較を行う．粒子登録法で近傍リストの半径を小さくした $\alpha = 0.2$ では，粒子間相互作用にかかる時間は 595.02 sec であり，リンクリスト法やハッシュ法で行った場合と比較して，約 31%に短縮されている．リンクリスト法やハッシュ法では，相互作用計算のときに参照する領域は，1 辺が影響半径の 3 倍の大きさの立方体であり，接触していない粒子が多く含まれる．一方，粒子登録法で参照する領域は，半径が影響半径よりわはずかに大きい球形なので，接触していない粒子の参照が少なくなる．そのため，接触していない粒子との無駄な計算が少なくなり，相互作用に要する時間が短縮されたと考えられる．しかし，粒子登録法は近傍粒子リストの作成に時間がかかるため，全体の計算時間はリンクリスト法とハッシュ法のほうが短い．

7.4　粒子登録法とリンクリスト法のハイブリット手法および粒子登録法とハッシュ法のハイブリット手法の性能評価

粒子登録法とハッシュ法のハイブリット手法における，近傍粒子リストの更新頻度の計算時間への影響を調べ，最適な更新頻度を決定するパラメータ α を設定する．$\alpha = 0.05$

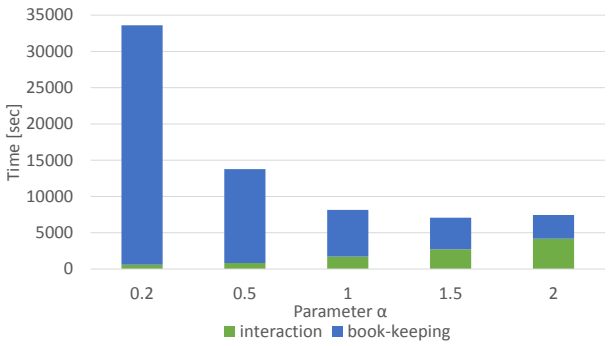

図 10　粒子登録法における近傍粒子リストの更新頻度による計算時間の違い
Fig. 10　How frequency of update affects calculation time in book-keeping method.

から $\alpha = 0.4$ まで変化させたときの計算時間を**図 11** に示す．緑色が粒子間相互作用の計算にかかった時間を，青色が粒子登録法の近傍粒子リストの作成にかかった時間を示す．図 10 の粒子登録法のみの結果と同様に，パラメータ α が小さいほど相互作用計算にかかる時間は短く，α が大きほど近傍粒子リストの作成にかかる時間が短い．図 10 の粒子登録法のみの場合は，近傍粒子リストの作成に相互作用計算の何倍もの時間がかかっているが，近傍リストの作成時にハッシュ法による近傍探索を加えることにより，リスト作成にかかる時間を大幅に削減できている．今回の検証では，パラメータ $\alpha = 0.1$ とした場合に計算時間が最も短いため，粒子登録法とハッシュ法のハイブリット手法では最適な α は 0.1 とする．粒子登録法とリンクリストを用いた方法を組み合わせた場合も同様の傾向が得られており，こちらも α の最適値は 0.1 とする．

更新頻度を決定するパラメータ α を 0.1 と設定した場合の計算時間の内訳を**表 4** に示す．表 4 の左側から，粒子間相互作用の計算，粒子登録法の近傍粒子リストの作成，リンクリストの作成，ハッシュ法，2 段 2 次ルンゲクッタ法による時間発展，全体の計算にかかった時間である．

表 3 のリンクリスト法とハッシュ法の場合と比べ，粒子登録法と組み合わせたハイブリット手法は，粒子間相互作用の計算にかかる時間を平均で 27%に短縮することが出来た．これは，周囲のセルに含まれる粒子から，粒子登録法でさらに自身の近傍の粒子を絞りこみ，相互作用計算を行ったためである．セル分割による近傍探索を近傍粒子リストの更新のときにのみ行うので，リンクリストの作成

Vol.2014-ARC-213 No.24
Vol.2014-HPC-147 No.24
2014/12/10

表 4　粒子登録法とリンクリスト法のハイブリット手法および粒子登録法とハッシュ法のハイブリット手法の計算時間

Table 4　Calculation time of book-keeping + linked-list method and book-keeping + hash method.

	interaction	book-keeping	linked-list	hash	update	total
book-keeping + linked-list	526.58	62.97	1.35	—	18.90	610.03
book-keeping + hash	525.00	81.03	—	2.84	19.20	628.29

unit : sec

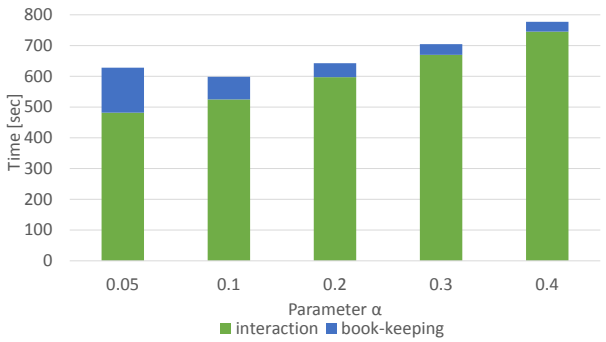

図 11　粒子登録法とハッシュ法のハイブリット手法における近傍粒子リストの更新頻度による計算時間の違い

Fig. 11　How frequency of update affects calculation time in book-keeping + hash method.

とハッシュ法にかかった時間が数秒程度になっている．計算全体としてはハイブリット手法を用いると，相互作用の計算が大幅に短縮されるので，リンクリスト法およびハッシュ法で計算した場合と比べて，計算時間を約 31%に削減することが出来た．

粒子登録法とリンクリスト法のハイブリット手法と粒子登録法とハッシュ法のハイブリット手法を比較すると，粒子登録法とリンクリスト法を用いたほうが，近傍粒子リストの作成にかかる時間を粒子登録法とハッシュ法を用いた場合の 77%に短縮することができている．7.2 節で示したように，周囲セルに含まれる粒子のインデックスの参照をハッシュ法よりもリンクリスト法のほうが高速に行えるためである．この差により全体の計算時間はリンクリスト法を用いたほうが 18.26 sec 短い．粒子登録法とリンクリスト法のハイブリット手法が本論文で実装した 5 つの近傍探索手法のなかで，最も計算時間が短かい．計算速度を求める場合は近傍探索手法に粒子登録法とリンクリスト法のハイブリット手法を用いることが最適であることがわかった．

8.　各近傍探索手法のメモリ使用量の比較

3 次元 DEM 計算によるダム崩壊シミュレーションを行ったときの，各近傍探索手法で確保したメモリ量を**表 5** に示す．左から粒子の物理量，リンクリスト法，ハッシュ法，粒子登録法のメモリ使用量である．粒子登録法の近傍粒子リストの更新パラメータ α は，計算速度が最も速かった最適値を用いており，粒子登録法のみの場合は $\alpha = 1.5$，ハイブリット手法では $\alpha = 0.1$ とし，近傍粒子リストの配列の大きさは，それぞれ 100 個，15 個の粒子インデックスが格納できる分確保している．格子サイズと格子数は，粒子登録法を用いない場合は，格子サイズは粒子直径と等しく，格子数は $201 \times 201 \times 201$ 個である．ハイブリット手法では，格子サイズは粒子直径の 1.1 倍であり，格子数は $182 \times 182 \times 182$ 個である．粒子物理量は倍精度で確保している．

粒子登録法のみの場合，近傍粒子リストの更新頻度を下げるためにリストの半径を大きくし，各粒子が多くの粒子のインデックスを格納出来るだけの配列を持つため，メモリ使用量が他の手法に比べて多い．リンクリスト法とハッシュ法を比較すると，ハッシュ法のほうがメモリ使用量が少なく，リンクリスト法の約 66%である．リンクリスト法とハッシュ法で計算時間に大きな差はないので，近傍粒子探索のメモリ使用量を抑えたい場合はハッシュ法を用いるのが良いと考えられる．

粒子登録法とリンクリスト法のハイブリット手法および粒子登録法とハッシュ法のハイブリット手法では，近傍粒子リストの半径を小さく設定しているため，粒子登録法のみの場合と比べて粒子登録法に必要なメモリが少ない．粒子登録法とリンクリスト法のハイブリット手法のメモリ使用量は合計で 132.2 MB であり，リンクリスト法の 1.6 倍である．粒子登録法とハッシュ法のハイブリット手法では 112.1 MB であり，ハッシュ法の 2.0 倍のメモリを使用している．

最も計算速度が速い粒子登録法とリンクリスト法のハイブリット手法では，粒子の物理量のメモリと合わせて 876.2 MB 必要となる．Tesla K20X はメモリが 6 GB なので，1 台に 500 万個以上の粒子を割り当てることができ，粒子登録法とリンクリスト法のハイブリット手法は大規模計算にも適用可能である．**図 12** は 500 万個の粒子を用いた DEM によるダム崩壊シミュレーションのスナップショットであり，Tesla K20X を用いて計算が可能であることを確認した．

DEM の影響半径は粒子直径であり，流体計算で用いられる SPH 法や MPS 法に比べて影響半径が小さく，影響半径内の粒子数が少ない．そのため，DEM の場合は粒子登

表 5　各近傍粒子探索手法で必要なメモリ量
Table 5　Size of memory usage in each neighboring list method.

	particle	book-keeping	linked-list	hash
book-keeping	744.0	404.0	—	—
lnked-list	744.0	—	85.0	—
hash	744.0	—	—	56.5
book-keeping + linked-list	744.0	64.0	68.2	—
book-keeping + hash	744.0	64.0	—	48.1

unit : MB

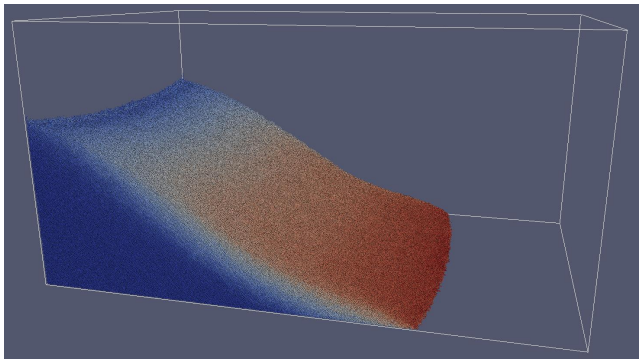

図 12　500 万個の粒子を用いた DEM によるダム崩壊
Fig. 12　Breaking dam simulation using 5 million particles.

録法で各粒子がもつ近傍粒子リストに，十数個の粒子インデックスを保持出来ればよい．SPH 法や MPS 法は影響半径内の粒子が DEM よりも多いので，粒子登録法を用いた場合，近傍粒子リストに 100 個以上の粒子インデックスを格納する必要があり，使用するメモリ量が DEM に比べて大幅に増える．SPH 法や MPS 法などの影響半径の大きい粒子方の計算での大規模計算に粒子登録法を用いた場合，近傍粒子リストによりメモリが枯渇してしまい，粒子登録法が実装できない可能性が考えられる．

9. 結言

近接相互作用に基づく粒子法のひとつである DEM の GPU での計算に，粒子登録法と，リンクリスト法と，ハッシュ法と，粒子登録法とリンクリスト法のハイブリット手法と，粒子登録法とハッシュ法のハイブリット手法の 5 つ近傍探索手法を実装し，各手法の違いによる計算時間およびメモリ使用量の比較を行った．はじめに，粒子の物理量のデータ構造による計算時間への影響を調べ，粒子データを SOA とすると，AOS とした場合と比べ，計算時間を短縮できることを確認した．近傍探索手法を比較する例題として，100 万個の粒子を用いた 3 次元 DEM によるダム崩壊シミュレーションをそれぞれの近傍探索手法で計算し，計算時間の測定を行った．その結果，粒子登録法は近傍粒子リストに N^2 の計算コストがかかるため，他の手法に比べて計算に時間が掛かることを確認した．リンクリスト法とハッシュ法を比較した結果，GPU 計算においてもリンクリスト法のほうが計算時間が短く，メモリ使用量はハッシュ法のほうが少ないことがわかった．粒子登録法とリンクリスト法のハイブリット手法が 5 つの近傍探索手法のなかで最も計算時間が短く，メモリ使用量はリンクリスト法の 1.6 倍，ハッシュ法の 2.3 倍で抑えられている．以上のことより，メモリに余裕がある場合は近傍探索手法に粒子登録法とリンクリスト法のハイブリット手法を，メモリ使用量を極力抑えたい GPU 計算のような場合はハッシュ法を実装するのが適切であることがわかった．

謝辞　本研究の一部は科学研究費補助金・基盤研究（S）課題番号 26220002「ものづくり HPC アプリケーションのエクサスケールへの進化」，科学技術振興機構 CREST「ポストペタスケール高性能計算に資するシステムソフトウェア技術の創出」から支援を頂いた．記して謝意を表す．

参考文献

[1] S Koshizuka and Y Oka. Moving-particle semi-implicit method for fragmentation of incompressible fluid. *Nuclear science and engineering*, Vol. 123, No. 3, pp. 421–434, 1996.
[2] Joseph J Monaghan. An introduction to SPH. *Computer physics communications*, Vol. 48, No. 1, pp. 89–96, 1988.
[3] Peter A Cundall and Otto DL Strack. A discrete numerical model for granular assemblies. *Geotechnique*, Vol. 29, No. 1, pp. 47–65, 1979.
[4] 小口かなえ, 澁田靖, 鈴木俊夫. GPU を用いた分子動力学法解析の高速化. 日本金属学会誌, Vol. 76, No. 7, pp. 462–467, 2012.
[5] Nicolin Govender, Daniel N Wilke, Schalk Kok, and Rosanne Els. Development of a convex polyhedral discrete element simulation framework for NVIDIA Kepler based GPUs. *Journal of Computational and Applied Mathematics*, Vol. 270, pp. 386–400, 2014.
[6] 西浦泰介, 阪口秀. GPU を用いた DEM の高速化アルゴリズム. 日本計算工学会論文集, Vol. 2010, No. 20100007, 2010.
[7] Takahiro Harada, Seiichi Koshizuka, and Yoichiro Kawaguchi. Smoothed particle hydrodynamics on GPUs. In *Computer Graphics International*, pp. 63–70. SBC Petropolis, 2007.
[8] 都築怜理, 青木尊之, 下川辺隆史. GPU スパコンにおける 1 億個のスカラー粒子計算の強スケーリングと動的負荷分散. 情報処理学会論文誌. コンピューティングシステム, Vol. 6, No. 3, pp. 82–93, sep 2013.
[9] José M Domínguez, Alejandro JC Crespo, Daniel Valdez-Balderas, Benedict D Rogers, and Moncho Gómez-

Gesteira. New multi-GPU implementation for smoothed particle hydrodynamics on heterogeneous clusters. *Computer Physics Communications*, Vol. 184, No. 8, pp. 1848–1860, 2013.

[10] 都築怜理, 青木尊之. GPU による大規模粒子法シミュレーションの実問題への適用. 計算工学講演会論文集 Proceedings of the Conference on Computational Engineering and Science, 第 19 巻, p. 5. 日本計算工学会, 2014.

[11] 平林久義, 佐藤雅弘. 線形リストを用いた粒子法の近傍粒子探索. 日本計算工学会論文集, Vol. 2010, No. 20100001, 2010.

[12] Simon Green. Particle simulation using cuda. *NVIDIA Whitepaper, December 2010*, 2010.