

「Atom クラス」の数式解説 (atom.cpp)

Open DEM Japan

2025年6月30日

粒子系を構成する Atom クラスは、個別要素法 (*Discrete Element Method, DEM*) において「粒子個々の状態変数」を格納・管理し、さらに入力データの読込や空間並べ替えなどの補助処理を行う。本節では、コード上の変数名に触れず、数式のみでアルゴリズムの核心を記述する。

*

1. 粒子識別子の拡張各粒子 i はタグ $t_i \in \mathbb{N}$ を持つ。タグ未設定粒子 ($t_i = 0$) が存在する場合、全並列プロセスで

$$T_{\max} = \max_j t_j \quad (1)$$

を得たのち、未設定粒子へ

$$t_i \leftarrow T_{\max} + 1, T_{\max} + 2, \dots \quad (2)$$

の順で一意に割当て。これは MPI の全域演算によって全プロセスの整合性を保証する。

*

2. タグの連続性検査総粒子数を N とするとき、

$$\min_i t_i = 1, \quad \max_i t_i = N \quad (3)$$

が同時に成り立つか判定し、不成立ならば I/O 例外を発生させる。

*

3. シミュレーション領域への再配置周期境界を持つ直方体領域で粒子座標 \mathbf{x}_i と画像番号 $\mathbf{n}_i \in \mathbb{Z}^3$ が与えられたとき、

$$\mathbf{x}_i \leftarrow \mathbf{x}_i + \mathbf{n}_i \odot \mathbf{L}, \quad \mathbf{L} = (L_x, L_y, L_z) \quad (4)$$

により原領域へ折り戻す (\odot は要素積)。斜格子系の場合は変換

$$\boldsymbol{\lambda}_i = \mathbf{H}^{-1} \mathbf{x}_i, \quad \mathbf{x}_i = \mathbf{H} \boldsymbol{\lambda}_i \quad (5)$$

を介して行う。ここで \mathbf{H} は格子行列。

*

4. 空間ソート用ビン分割バウンディングボックス $\mathbf{b}_{lo}, \mathbf{b}_{hi} \in \mathbb{R}^3$ を h 間隔で分割する。軸方向ビン数は

$$n_\alpha = \max(1, \lfloor (b_{hi,\alpha} - b_{lo,\alpha})/h \rfloor), \quad \alpha \in \{x, y, z\}. \quad (6)$$

粒子 i を

$$k_\alpha(i) = \text{clamp}\left(\left\lfloor (x_{i,\alpha} - b_{\text{lo},\alpha}) \frac{n_\alpha}{b_{\text{hi},\alpha} - b_{\text{lo},\alpha}} \right\rfloor, 0, n_\alpha - 1\right) \quad (7)$$

で求められるビン (k_x, k_y, k_z) に割当てる. 全ビン数は

$$n_{\text{bin}} = n_x n_y n_z. \quad (8)$$

*

5. 並べ替えパーミュテーション粒子列 $\{1, \dots, N_{\text{loc}}\}$ とビン内リンクリストを用いて所望の順序 π を構成する. 交換操作は

$$\mathbf{q}_{\pi(i)} \leftrightarrow \mathbf{q}_{\pi(j)}, \quad (9)$$

ただし \mathbf{q} は粒子状態ベクトル. 任意の置換は高々 1 つの一時領域を用いて巡回置換ごとに実装できる.

*

6. 質量一貫性判定粒子型 s の半径 r_i について

$$\exists i, j \text{ with } r_i \neq r_j \implies \text{不整合}. \quad (10)$$

MPI 全域演算で“真”が 1 つでも得られた場合, エラーとする.

*

7. 形状同一性判定 (楕円体例) 半軸長 $\mathbf{a}_i = (a_{i1}, a_{i2}, a_{i3})$ について

$$\exists i, j \text{ with } \mathbf{a}_i \neq \mathbf{a}_j \implies \text{不整合}. \quad (11)$$

*

8. ビンサイズ決定 (CUDA 最適化) 平均粒子数を $n_0 = 3000$ とし, 三次元系なら

$$h = \left(\frac{V_{\text{dom}}}{N n_0}\right)^{1/3}, \quad V_{\text{dom}} = (b_{\text{hi},x} - b_{\text{lo},x})(b_{\text{hi},y} - b_{\text{lo},y})(b_{\text{hi},z} - b_{\text{lo},z}). \quad (12)$$

二次元系では立方根を平方根に置き換える.

*

9. メモリ使用量状態変数総メモリは

$$M = \sum_{v \in \mathcal{V}} N_{\text{max}} \text{sizeof}(v) + M_{\text{map}} + M_{\text{bin}} + M_{\text{perm}}, \quad (13)$$

ただし \mathcal{V} は登録済みベクトル集合. マップは

$$M_{\text{map}} = \begin{cases} (T_{\text{max}} + 1) \text{sizeof}(\text{int}), & \text{array 型} \\ n_{\text{bucket}} \text{sizeof}(\text{int}) + n_{\text{hash}} \text{sizeof}(\text{HashElem}), & \text{hash 型.} \end{cases} \quad (14)$$

*

10. コールバック管理外部機能 F_k のコールバック集合を

$$\mathcal{C}_\ell = \{c_k \mid \text{種別} = \ell, k = 1, \dots, N_\ell\} \quad (\ell = 0, 1, 2), \quad (15)$$

とする. 削除対象 F_{k^*} が与えられた際,

$$\mathcal{C}_\ell \leftarrow \mathcal{C}_\ell \setminus \{c_{k^*}\} \quad (16)$$

を全 ℓ について実行し, 後続インデクスを減算更新する.

以上により, Atom クラスの主要機能はタグ操作 [(1)–(3)], 周期境界処理 [(4)], 空間ソート [(6)–(9)], 物性値一貫性検査 [(10), (11)], GPU 最適化ビンサイズ選定 [(12)], およびメモリ・コールバック管理 [(13)–(16)] に集約できることがわかる.