

煉瓦型通信アルゴリズムの数式解説 (comm_brick.cpp)

Open DEM Japan

2025年6月30日

本稿では、離散要素法 (DEM) コード LIGGGHTS の通信クラス CommBrick が実装している煉瓦型 (Brick) 空間分割通信アルゴリズムを、数式を用いて逐次的に解説する。以下では変数名の説明は割愛し、アルゴリズムの数理構造に焦点を当てる。

■1. ゴースト幅の決定 粒子間相互作用の最遠距離を

$$c_{\max} = \max(c_{\text{neigh}}, c_{\text{user}}) \quad (1)$$

とすると、 d 軸方向 ($d = x, y, z$) のゴースト幅は

$$g_d = \begin{cases} c_{\max}, & \text{直交格子の場合,} \\ c_{\max} \|\mathbf{h}_d^{-1}\|_2, & \text{傾斜 (triclinic) 格子の場合,} \end{cases} \quad (2)$$

で与えられる。ここで \mathbf{h}^{-1} は逆格子行列である。

■2. 必要通信階層の深さ 全プロセスグリッドを $\mathbf{N} = (N_x, N_y, N_z)$ 、計算領域の長さを $\mathbf{L} = (L_x, L_y, L_z)$ とする。式 (2) より、

$$n_d = \left\lceil \frac{g_d N_d}{L_d} \right\rceil \quad (d = x, y, z) \quad (3)$$

を計算し、非周期境界がある場合は $n_d \leq N_d - 1$ と切り詰める。この n_d が “何プロセス先までゴースト粒子を受信するか” を決定する。

$$n_{\max} = n_x + n_y + n_z$$

とにおいて、実際に行う通信回数は

$$n_{\text{swap}} = 2 n_{\max} \quad (4)$$

となる。

■3. スワップごとの送信領域 各スワップ s に対し、送信スラブの境界 ($s_{\text{lo}}^{(s)}, s_{\text{hi}}^{(s)}$) は

$$s_{\text{lo}}^{(s)} = \begin{cases} -\infty & \text{最外層送信で左方向へ送る場合} \\ \frac{\ell_{\text{lo}} + \ell_{\text{hi}}}{2} & \text{それ以外} \end{cases}, \quad (5)$$

$$s_{\text{hi}}^{(s)} = \begin{cases} \ell_{\text{lo}} + g_d & \text{左方向へ送る場合} \\ +\infty & \text{最外層送信で右方向へ送る場合,} \\ \frac{\ell_{\text{lo}} + \ell_{\text{hi}}}{2} & \text{右方向 2 層目以降} \end{cases}, \quad (6)$$

となる。 $\ell_{\text{lo/hi}}$ は自分の部分領域端である。

■4. PBC 係数 周期境界を持つ場合、`pbs_flag` と 6 成分ベクトル $\mathbf{p} = (p_x, p_y, p_z, p_{yz}, p_{xz}, p_{xy})$ を

$$p_d = \begin{cases} +1 & \text{左端から粒子を受け取る場合} \\ -1 & \text{右端へ粒子を送る場合} \\ 0 & \text{境界を跨がない場合} \end{cases}$$

で設定し、位置ベクトルの再周期化

$$\mathbf{x} \leftarrow \mathbf{x} + p_x L_x \hat{\mathbf{e}}_x + p_y L_y \hat{\mathbf{e}}_y + p_z L_z \hat{\mathbf{e}}_z \quad (7)$$

を行う。傾斜格子ではオフダイアゴナル成分 (p_{yz}, p_{xz}, p_{xy}) も同時に加算する。

■5. 境界粒子の判定 単一カット方式では、粒子 i がスワップ s で送信対象となる条件を

$$x_{i,d} \notin [s_{\text{lo}}^{(s)}, s_{\text{hi}}^{(s)}], \quad (8)$$

半径最適化を有効にした場合は

$$x_{i,d} \pm r_i \notin [s_{\text{lo}}^{(s)}, s_{\text{hi}}^{(s)}] \quad (9)$$

で判定する。

■6. 送受信バッファの構築と通信 送信粒子数を $n_s^{(s)}$ とすると、送信バイト長は

$$b_s^{(s)} = n_s^{(s)} m_{\text{border}}, \quad (10)$$

ここで m_{border} は粒子 1 個あたりのデータサイズである。隣接プロセス p_{send} , p_{recv} と

$$\text{MPI_Sendrecv}(b_s^{(s)}, p_{\text{send}}; b_r^{(s)}, p_{\text{recv}}) \quad (11)$$

を行い、受信粒子数 $n_r^{(s)}$ を得る。

■7. 交換後の粒子総数 交換完了時点で、各プロセスのローカル粒子数

$$n'_{\text{local}} = n_{\text{own}} + n_{\text{ghost}}$$

を集約し、

$$N_{\text{atoms}} = \sum_{p=0}^{N_{\text{proc}}-1} n'_{\text{local}}(p) \quad (12)$$

が全系粒子数として更新される。

■8. 力の逆通信 逆通信では、ゴースト粒子に作用した力 $\mathbf{f}_i^{(\text{ghost})}$ を所有プロセスに返送する。演算順序を反転させて

$$\mathbf{f}_i^{(\text{own})} \leftarrow \mathbf{f}_i^{(\text{own})} + \sum_{s=n_{\text{swap}}-1}^0 \mathbf{f}_i^{(s)} \quad (13)$$

を加算することで、運動方程式

$$m_i \frac{d^2 \mathbf{x}_i}{dt^2} = \mathbf{f}_i^{(\text{own})} \quad (14)$$

が整合的に解かれる。

■9. メモリ使用量 主要バッファのメモリ使用量は

$$M = \sum_{s=0}^{n_{\text{swap}}-1} \left(\underbrace{n_s^{(s)}}_{\text{インデックス}} + \underbrace{b_s^{(s)}}_{\text{送信}} + \underbrace{b_r^{(s)}}_{\text{受信}} \right) + M_{\text{static}} \quad (15)$$

で評価され、実装では安全係数 $\text{BUFFACTOR}=1.5$ を乗じて再割り当てを抑制している。

■10. まとめ 以上の式 (1)–(15) により、CommBrick クラスの通信パターン生成から粒子交換・力逆通信に至る一連の操作が定式化された。本アルゴリズムは

- ゴースト幅を基準にした必要通信層深さの決定 (式 (3))
- 各層ごとの送信領域と PBC 係数設定 (式 (6),(7))
- 粒子半径を考慮した境界判定最適化 (式 (9))
- MPI Sendrecv を用いた二方向通信 (式 (11))
- 力の逆通信による運動方程式の一貫性維持 (式 (13)–(14))

で構成されており、煉瓦状に分割された並列空間上で最小限の通信コストで粒子移動を正確に管理する。