

FDPSの概要説明

岩澤全規

松江工業高等専門学校

2025/09/12 FDPS講習会

FDPSとは

- Framework for Developing Particle Simulator
- 大規模並列シミュレーションコードの開発を支援するフレームワーク
- 重力N体、SPH、分子動力学(MD)、個別要素法(DEM)、etc...
- 支配方程式

$$\frac{d\vec{u}_i}{dt} = \vec{g} \left(\sum_j^N \vec{f}(\vec{u}_i, \vec{u}_j), \vec{u}_i \right)$$

粒子データのベクトル

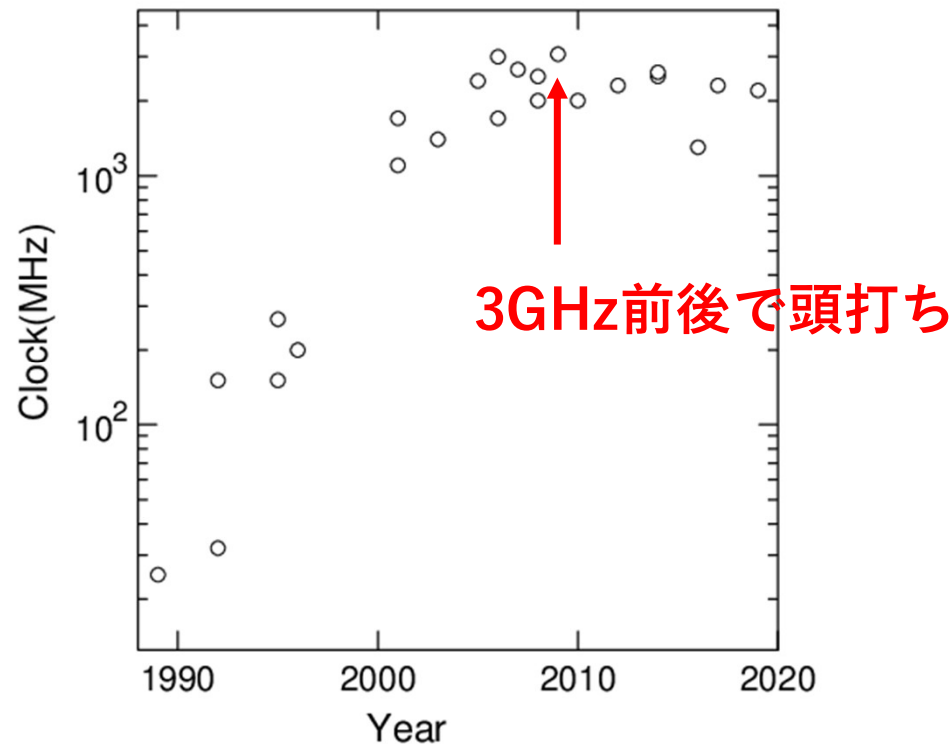
粒子間相互作用を表す関数

粒子の持つ物理量をその導関数に変換する関数

The diagram shows the governing equation for particle simulation: $\frac{d\vec{u}_i}{dt} = \vec{g} \left(\sum_j^N \vec{f}(\vec{u}_i, \vec{u}_j), \vec{u}_i \right)$. Red circles highlight the terms $\vec{g}$, $\vec{f}$, and $\vec{u}_i$ within the function arguments. Red arrows point from Japanese text labels to these circled terms: '粒子データのベクトル' (Vector of particle data) points to $\vec{u}_i$; '粒子間相互作用を表す関数' (Function representing particle-particle interaction) points to $\vec{f}$; and '粒子の持つ物理量をその導関数に変換する関数' (Function converting physical quantities of particles into their derivatives) points to $\vec{g}$.

大規模並列粒子 シミュレーションの必要性

- 逐次計算の速度はもう速くならない



大規模並列粒子シミュレーションの困難

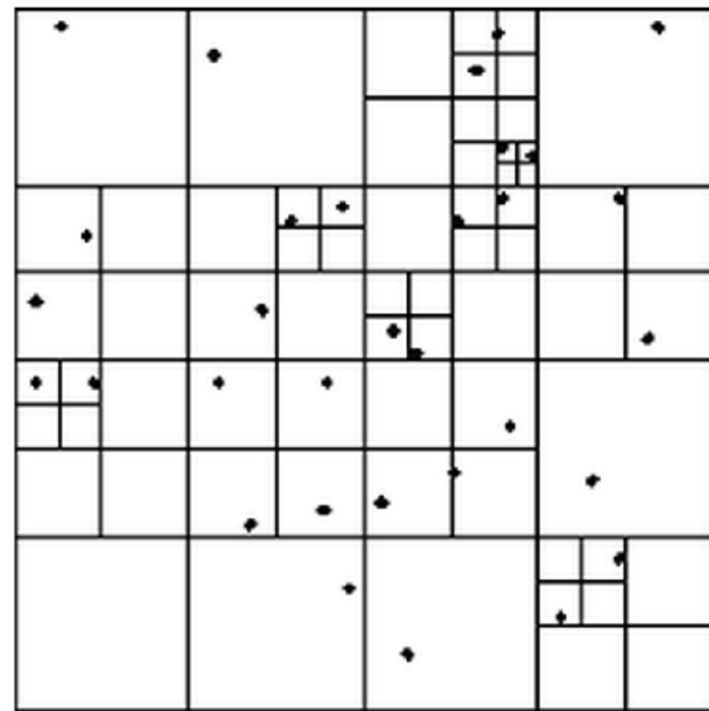
- 分散メモリ環境での並列化
 - 計算領域の分割と粒子データの交換
 - 相互作用計算のための粒子データの交換
- 共有メモリ環境での並列化
 - 相互作用計算の負荷分散
- 1コア内での並列化
 - SIMD演算器の有効利用

実は並列でなくとも、、、

- キャッシュメモリーの有効利用
- ツリー構造の構築

$$\frac{d\vec{u}_i}{dt} = \vec{g} \left(\sum_j^N \vec{f}(\vec{u}_i, \vec{u}_j), \vec{u}_i \right)$$

Nを小さい数に減らす方法



FDPSで困難を解決

- 分散メモリ環境での並列化
 - 計算領域の分割と粒子データの交換
 - 相互作用計算のための粒子データの交換
- 共有メモリ環境での並列化
 - 相互作用計算の負荷分散
- キャッシュメモリの有効利用
- Tree構造による粒子分布の管理
- 1コア内での並列化
 - SIMD演算器の有効利用

FDPS

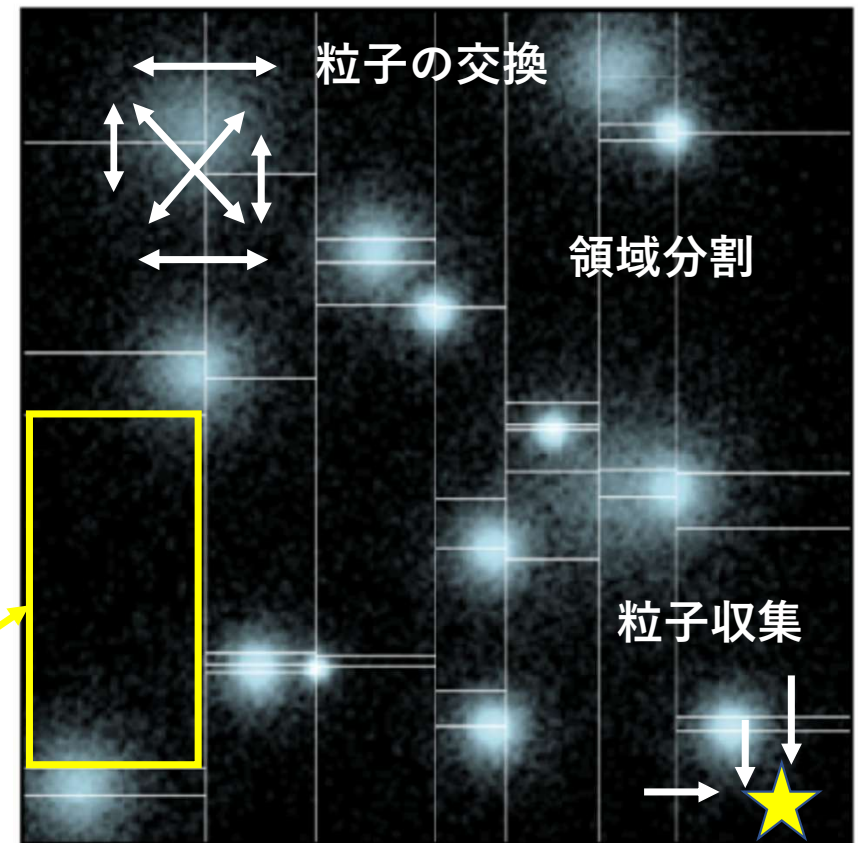
PIKG (牧野さん講義)

粒子シミュレーションの手順

FDPS

- 計算領域の分割
 - 粒子データの交換
 - 相互作用計算のための粒子データの収集
 - 実際の相互作用の計算
-
- 粒子の軌道積分

1つのプロセスが担当する領域



FDPSの実装方針(1)

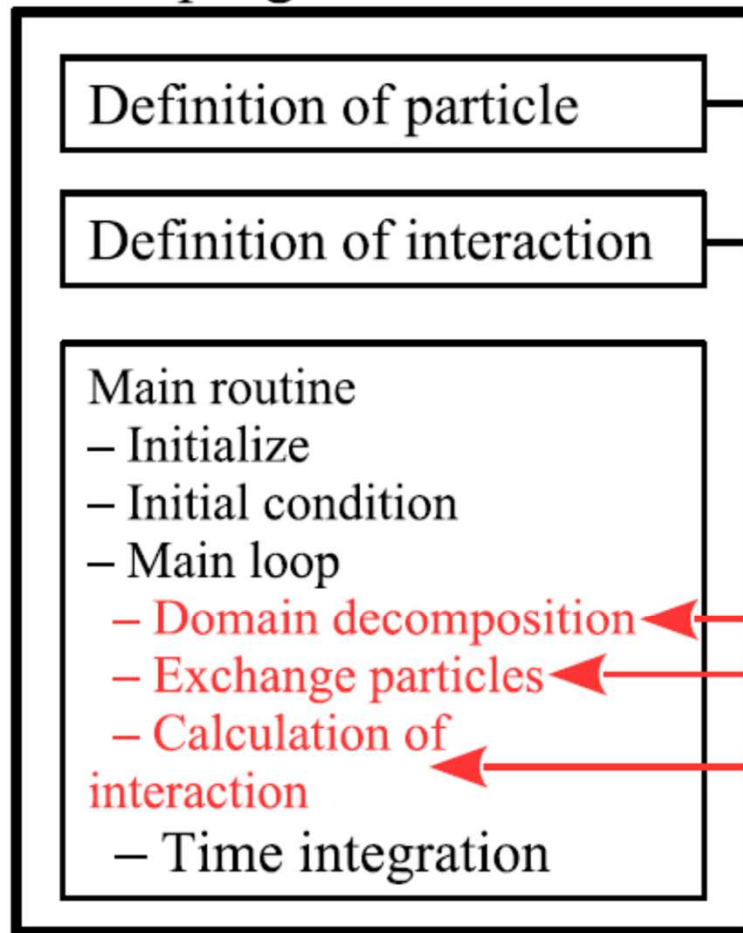
- 内部実装の言語としてC++を選択
 - 高い自由度
 - 粒子データの定義にクラスを利用
 - 相互作用の定義に関数ポインタ関数オブジェクトを利用
 - 高い性能
 - 上のクラスや相互作用関数を受け取るためにテンプレートを利用
 - コンパイル時に静的にコード生成するため

FDPSの実装方針(2)

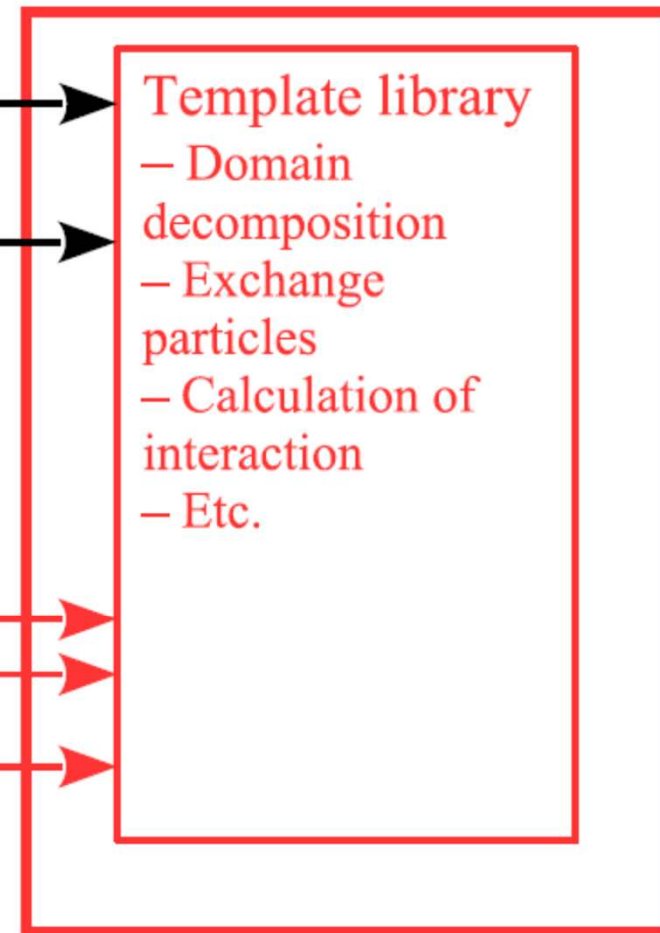
- 並列化
 - 分散メモリ環境(ノード間) : MPI
 - 共有メモリ環境(ノード内) : OpenMP

FDPSの基本設計

User program



FDPS

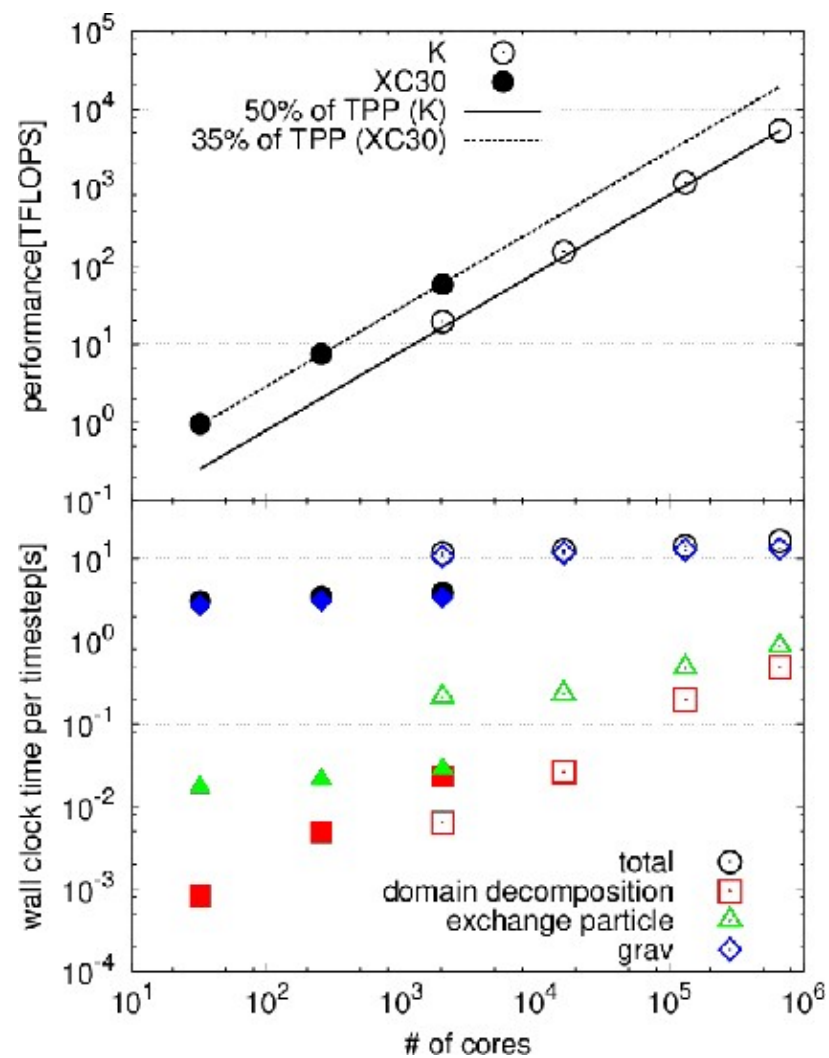
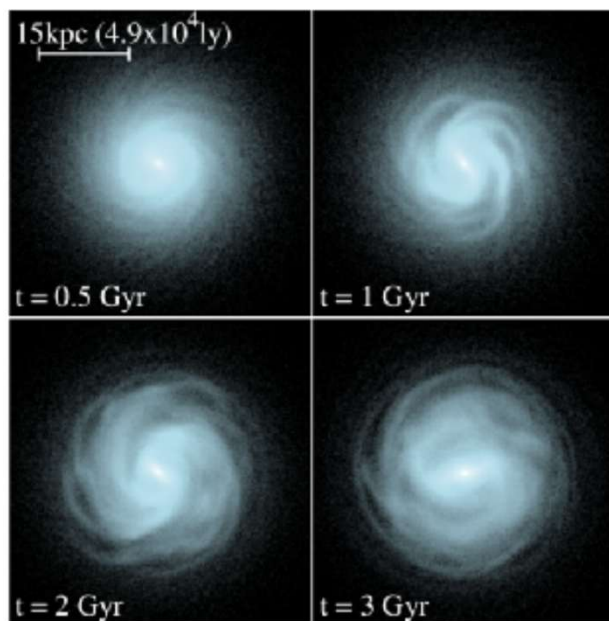


重要なポイント

- ユーザーはMPIやOpenMPによる並列化を考えなくてよい。
 - 相互作用関数の実装について
 - 2重ループ：複数の粒子に対する複数の粒子からの作用を計算
 - チューニングが必要
- ※PIKGを使えば最適化された相互作用関数を生成してくれる。

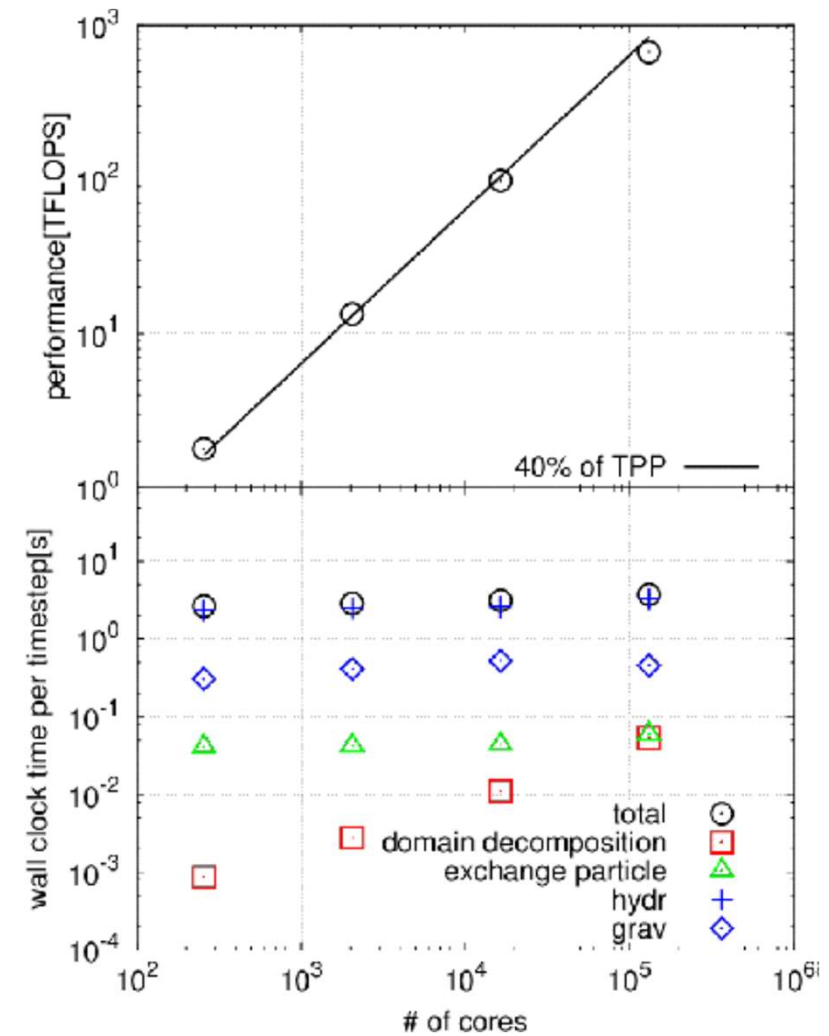
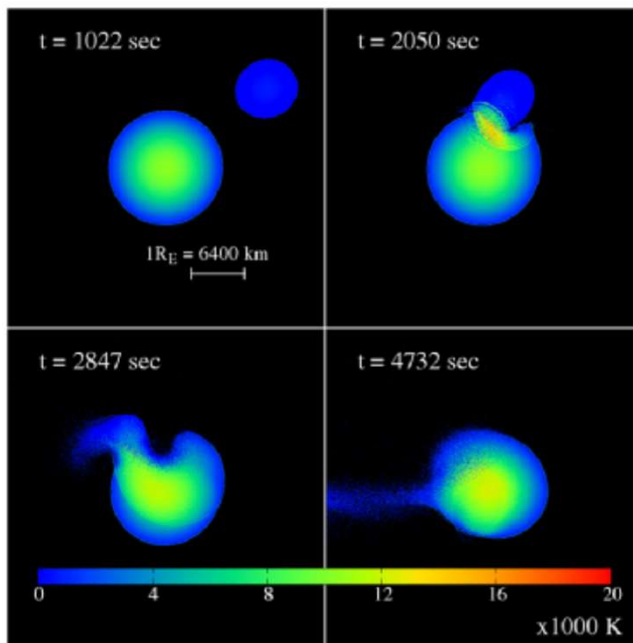
性能(N体)

- 円盤銀河
- 粒子数: $2.7 \times 10^5 / \text{core}$
- 精度: $\Theta = 0.4$ 四重極
- 京コンピュータ, XC30



性能(SPH)

- 巨大衝突シミュレーション
- 粒子数: $2.0 \times 10^4/\text{core}$
- 京コンピュータ



FDPSのリリースノート

- 2012年11月 FDPSの開発開始
- 2015年3月 FDPS Ver. 1.0
- 2016年1月 FDPS Ver. 2.0
 - アクセラレータ利用のために、Multiwalk法の実装
- 2016年12月 FDPS Ver. 3.0
 - Fortran Interfaceの実装
- 2017年11月 FDPS Ver. 4.0
 - SPH法やMD計算等で計算を高速化するために、相互作用リスト再利用のアルゴリズムの実装
- 2018年11月 FDPS Ver.5.0
 - C Interfaceの実装
- 2020年8月 FDPS Ver.6.0
 - PIKGの実装
- 2021年8月 FDPS Ver.7.0
 - 極座標でのツリー構築をサポートする機能の実装
- 2025年9月 FDPS Ver.8.0
 - PMMM法の導入
- 202x年y月 FDPS Ver.9.0
 - GPU対応

まとめ

- FDPSは大規模並列粒子シミュレーションコードの開発を支援するフレームワーク
- FDPSのAPIを呼び出すだけで粒子シミュレーションを並列化
- N体コードを100行程度で記述
- 京コンピュータで理論ピーク性能の40、50%の性能を達成