N 体計算の現状と将来

牧野淳一郎
東京大学理学系研究科天文学専攻

発表構成

1. なんのために重力多体系の数値計算をするか？
2. どんな問題が起きるか？
3. 役に立つ計算をするにはどうすればいいか？

なんのために計算するのか？

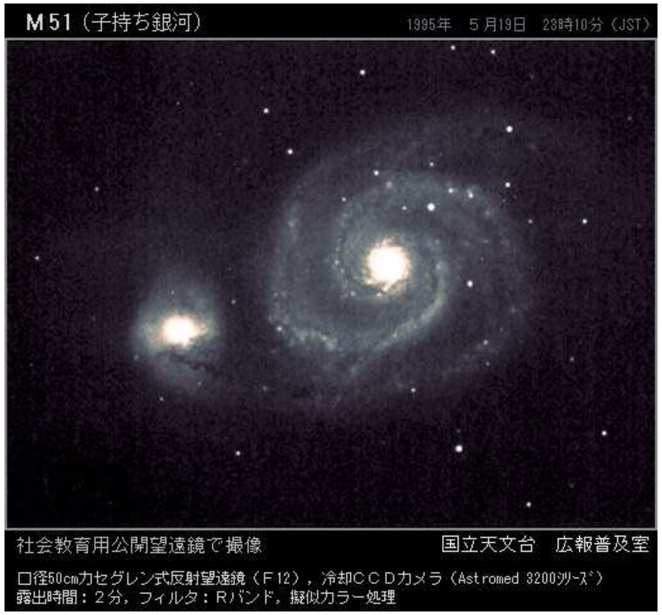
- 重力多体系とはどんなものか？
- 数値シミュレーションでなにをわかろうとするのか？
- で、問題は？

重力多体系

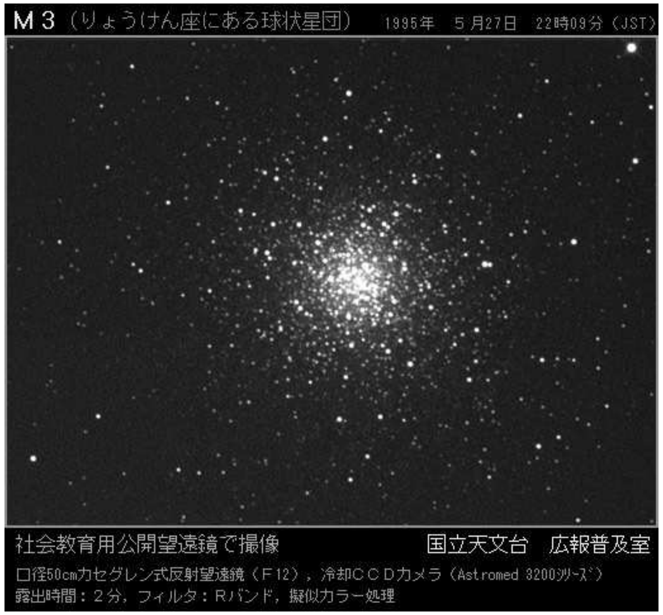
- そもそもどんなもの？
- 基礎方程式
- 無衝突系/衝突系
- 2 体緩和

そもそもどんなもの？

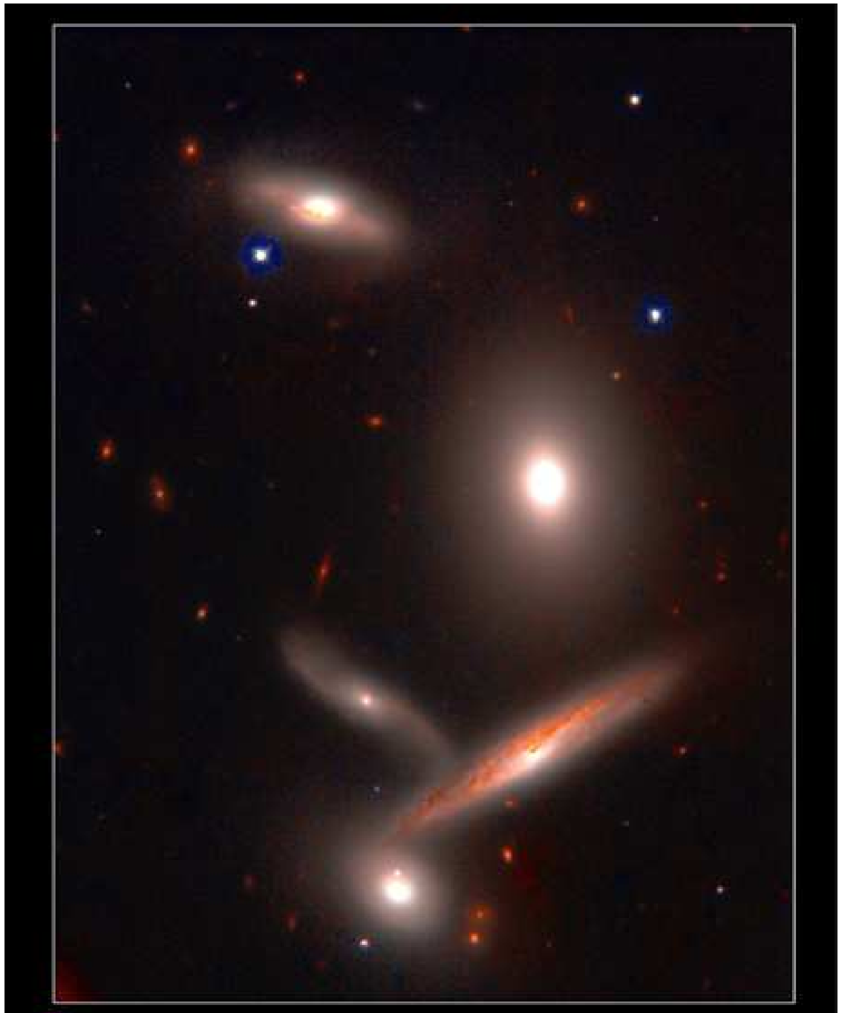
銀河


M 51（子持ち銀河）
1995年　５月19日　23時10分（JST）
社会教育用公開望遠鏡で撮像
国立天文台　広報普及室
口径50cmカセグレン式反射望遠鏡（Ｆ12），冷却ＣＣＤカメラ（Astromed 3200ｼﾘｰｽﾞ）
露出時間：２分，フィルタ：Ｒバンド，擬似カラー処理

球状星団


M 3（りょうけん座にある球状星団）
1995年　５月27日　22時09分（JST）
社会教育用公開望遠鏡で撮像
国立天文台　広報普及室
口径50cmカセグレン式反射望遠鏡（Ｆ12），冷却ＣＣＤカメラ（Astromed 3200ｼﾘｰｽﾞ）
露出時間：２分，フィルタ：Ｒバンド，擬似カラー処理

銀河群



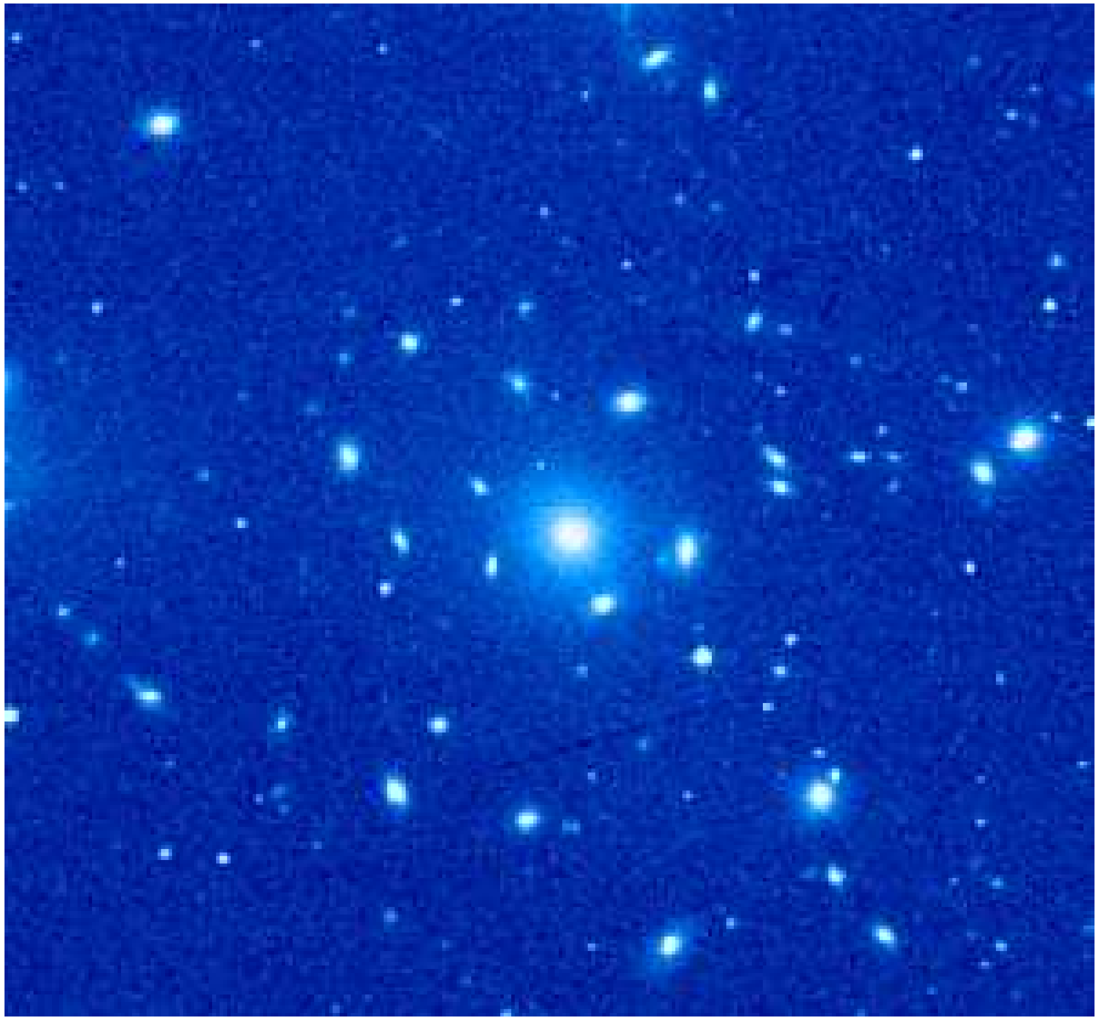
Hickson Compact Group 40

Subaru Telescope, National Astronomical Observatory of Japan

CISCO (J & K')

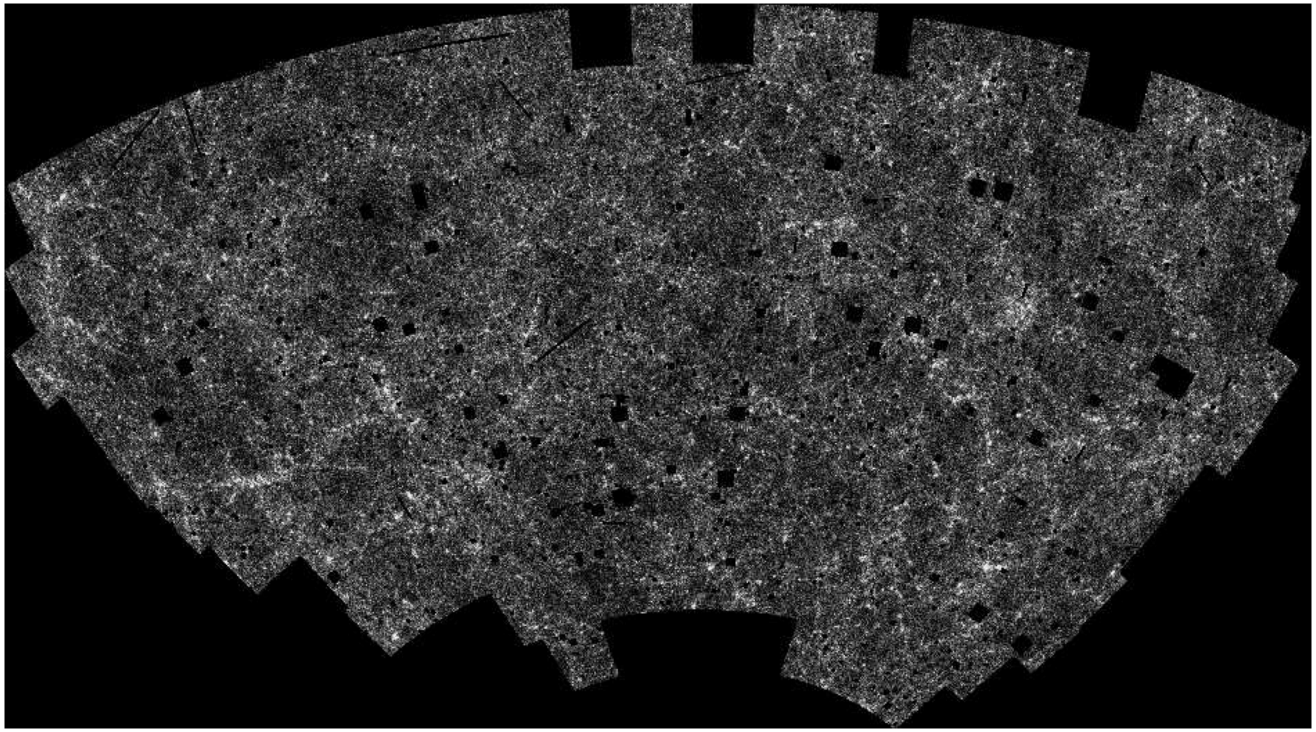
January 28, 1999

銀河団



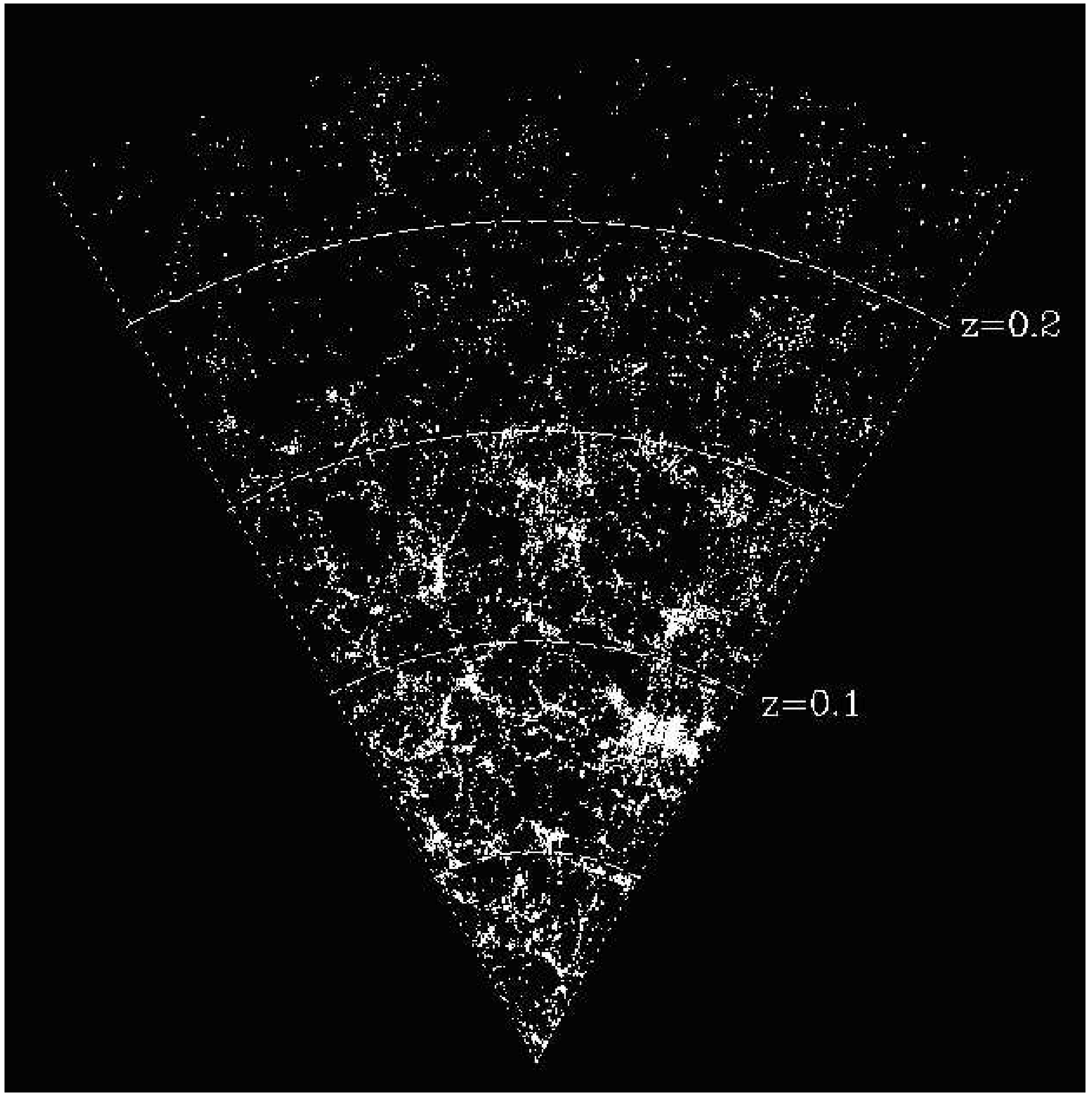
http://antwrp.gsfc.nasa.gov/apod/ap950917.html

大規模構造 (天球面)



http://www-astro.physics.ox.ac.uk/~wjs/apm_grey.gif

大規模構造 (距離情報あり) — SDSS スライス



重力多体系の基礎方程式

もとの方程式自体はもちろん、各粒子の運動方程式

$$\frac{d^2\boldsymbol{x}_i}{dt^2} = \sum_{j\neq i} Gm_j \frac{\boldsymbol{x}_j - \boldsymbol{x}_i}{|\boldsymbol{x}_j - \boldsymbol{x}_i|^3}, \tag{1}$$

数値シミュレーションでなにをわかろうとするのか

(極めて特別な場合を除いては)

運動方程式は解析的に解けない。

もちろん、解ける場合や、近似的に解ける場合というのはいっぱいある。

しかし、解析的には解けなくたって、計算機を使えば数値解は求められる。

計算すればなんでもわかるか？

原理的には答は YES。

現実の問題としては NO。

- 計算機の能力
- 素過程の理解

例:銀河形成・進化

典型的な銀河: 1000億個くらいの星、それと同程度の質量のガス、その数倍の質量の「ダークマター」からなる(ということになっている)

- 1000億体問題は今の計算機では全く無理
- 1000億体問題が解けても、それだけでは駄目
  - ガスから星ができる過程
  - 超新星爆発等で星がガスに戻る過程
  - ダークマターはどうするか

とりあえず、素過程が「わかっている」ものについては原理的には計算できる。まずはそういう前提で話を進めてみる。

本当に計算できるか？

銀河の例にあったように普通は「できない」

銀河の星の数: $\mathbf{10^{11}}$

多体計算で扱える粒子数:多くて $\mathbf{10^8}$

- なるべく多くの粒子を扱うにはどうするか
- 少ない粒子で必要な答を得るにはどうするか (そんなことができるか)

が問題。

粒子数と「答の正しさ」

原理的には、実際の系と同じ粒子数が扱えれば(重力多体系と近似していい範囲で)「正しい」

では、少ない粒子数で計算した時にどういう意味で正しくないのか？

少ない粒子数で計算するとはそもそもどういうことか？

少ない粒子数での計算

少なくとも以下の 3 種類

- 実質粒子数無限大の系を離散化
- 本当に有限個の粒子からできている系を、もっと少ない系でごまかす。
- 小さな部分系を切り出す

それぞれ別の問題がある。主に上の 2 つを考える。

- 空間分解能
- 数値拡散 (2 体緩和)
- タイムスケールの不一致
- 境界の影響

粒子数無限大の系

- 宇宙論
- 銀河形成

CDM を信じるなら (というか、barionic とか self-interacting とかいったものでも考えない限り) 実効的な粒子数は無限大

N 体計算: 6 次元空間の中での分布関数を、離散近似することに相当

- 初期ゆらぎの成長: 離散化誤差 (理屈は簡単)
- virialize した系: 主に 2 体緩和・ポテンシャルの誤差

2体緩和

粒子数無限大: 無衝突ボルツマン方程式

$$\frac{\partial f}{\partial t} + v \cdot \nabla f - \nabla \Phi \cdot \frac{\partial f}{\partial v} = \frac{\partial f}{\partial t}|_{coll}, \tag{2}$$

f:6次元位相空間での分布関数 Φ :重力ポテンシャル, 以下のポアソン方程式の解

$$\nabla^2 \phi = -4\pi G \rho. \tag{3}$$

ここで、 G は重力定数であり、 ρ は空間での質量密度

$$\rho = m \int dv f, \tag{4}$$

$\frac{\partial f}{\partial t}|_{coll}$: 衝突項。 $N \rightarrow \infty$ では 0 になる。

2体緩和

有限粒子数の自己重力多体系の進化

最初は力学平衡にない $\longrightarrow$ まず力学平衡に

有限粒子数: 真の力学平衡というものはない。

有限の質量をもった各粒子が系の中を運動するに従って、ポテンシャルが必ず変化する。この変化によって各粒子の軌道も変化することになる。 $=2$体緩和

緩和時間

適当に近似すると

$$< \Delta\theta^2 > \sim Gnv^{-3}m^2 \log(R/r_0) \tag{5}$$

R はシステムの大きさ、 r_0 は「大きく曲がる」ためのインパクトパラメータの値で、$b_0 = Gm/v^2$ の程度。

タイムスケール:

$$t_\theta \sim \frac{v^3}{Gnm^2 \log R/r_0} \tag{6}$$

多体系の緩和時間

質量密度一定の場合:タイムスケールが各粒子の質量に比例

質量 M、特徴的な半径（ビリアル半径か何か）R、粒子数 N とすれば、緩和のタイムスケールは

$$t_\theta \sim \frac{N}{\log N} t_d \tag{7}$$

となる。熱平衡 (等分配) にいく時間スケールも同じオーダー

注意:上の緩和時間は上限

- 密度が高い中心 (ρ に比例)
- 速度分散の小さい円盤 (v^3 に比例)

ずっと短くなる

実例

宇宙論的 N 体計算で銀河団のダークマター分布がどうなるか調べよう！— Navarro, Frenk, White (1996, 1997)

「高分解能」N 体計算でなにが出来るか調べた

結果：“universal profile”

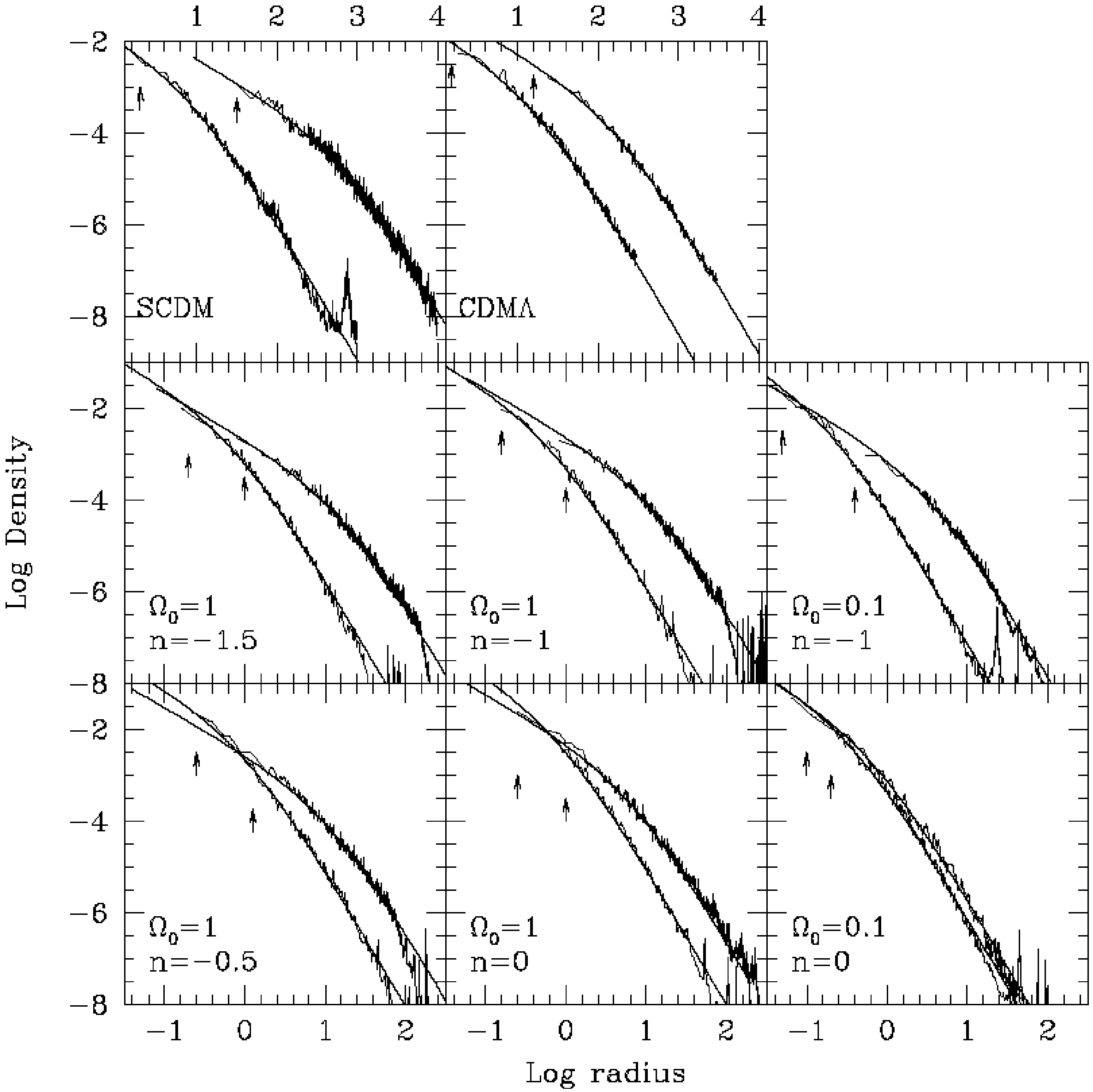
- CDM シナリオによる構造形成を考えた時、CDM が作る自己重力系 (ガスとか星を考えない) は

$$\rho \propto \frac{1}{r_*(1 + r_*)^2}$$

  の形に書ける

- この形はユニバーサルで、例えば一つの銀河でも銀河団でも同じである (NFW 1997)

NFW の結果



1
2
3
4
1
2
3
4
−2
−4
−6
−8
SCDM
CDMΛ
Log Density
Ω_0=1
n=−1.5
Ω_0=1
n=−1
Ω_0=0.1
n=−1
Ω_0=1
n=−0.5
Ω_0=1
n=0
Ω_0=0.1
n=0
−1
0
1
2
Log radius

NFW を信用していいか？

NFW の計算：ハロー一つに典型的には 10^4 体程度

問題点：中心部での二体緩和時間が宇宙年齢よりはるかに短い

緩和時間を「十分長く」した計算は出来ないか？

とにかく粒子数を増やす — **Fukushige and Makino 1997 (ApJ 477, L9)**

- NFW がいっているような中心で $1/r$ になる分布にはならない
- $\rho \sim r^{-1.4}$ 程度になった
- 信用できるのは 1-2kpc まで

Fukushige & Makino の結果

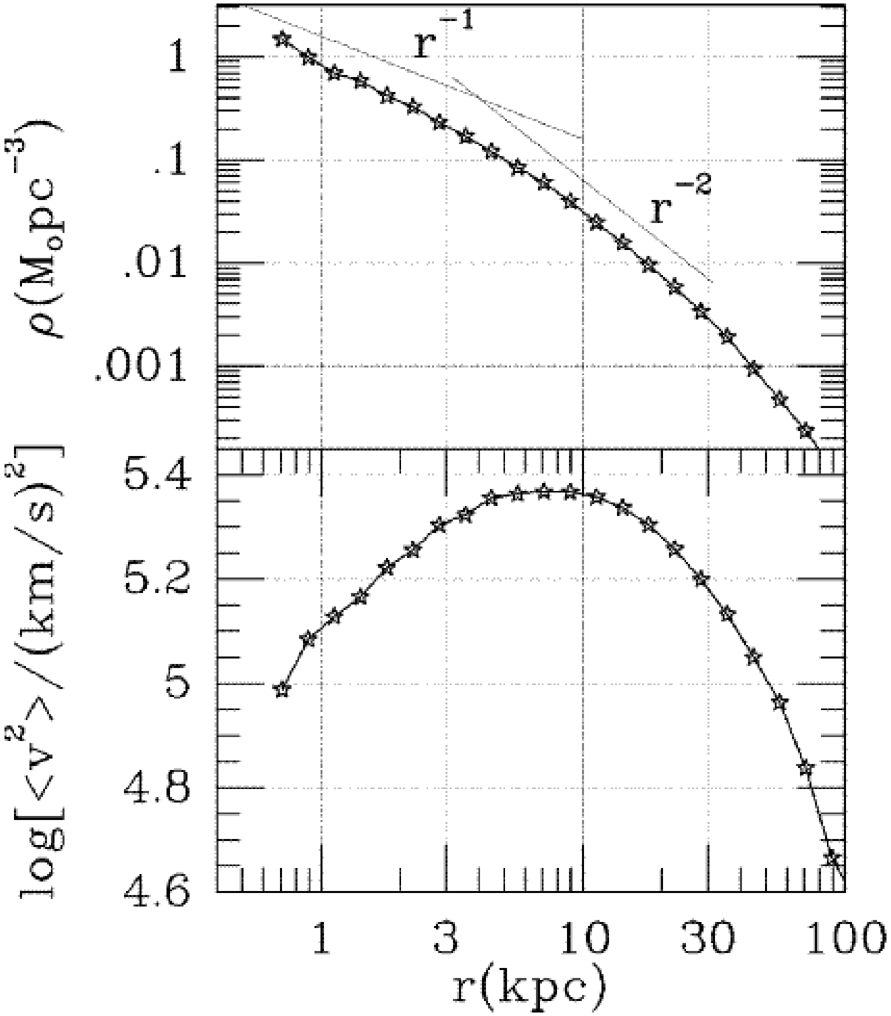


FIG. 2.—Density and temperature structures of the halo at $z = 1.8$ in the *N*-body simulation shown in Fig. 1. The position of the center of the halo was determined using potential minimum and averaged physical values over each spherical shell.

NFWの結果の解釈

- 「universal」になったのは数値計算の誤差のためであった（すべての計算で粒子数が同じ程度、ソフトニングパラメータも同じ程度だったので、結果がその2つで決まっていた）。

ソフトニング

$$\phi_{\mathrm{soft}} = -\frac{1}{\sqrt{\boldsymbol{r}^2 + \boldsymbol{\epsilon}^2}} \tag{8}$$

$\boldsymbol{O}\left(\frac{\boldsymbol{\epsilon}^2}{\boldsymbol{r}^2}\right)$ の誤差がポテンシャルにある。

経験的な規則

(Fukushige and Makino 2001)

- $t < t_{rel}(r)/3$
- $\Delta t < t_{dyn}/40$ (但し、可変でない時のみ)
- ϵ の 5 倍くらいの半径

なら、 log-log プロットをしていい程度には信用できる。

最近の結果

Fukushige and Makino (2001, astro-ph/0108014, 0008104)

500 万粒子くらいまでの計算

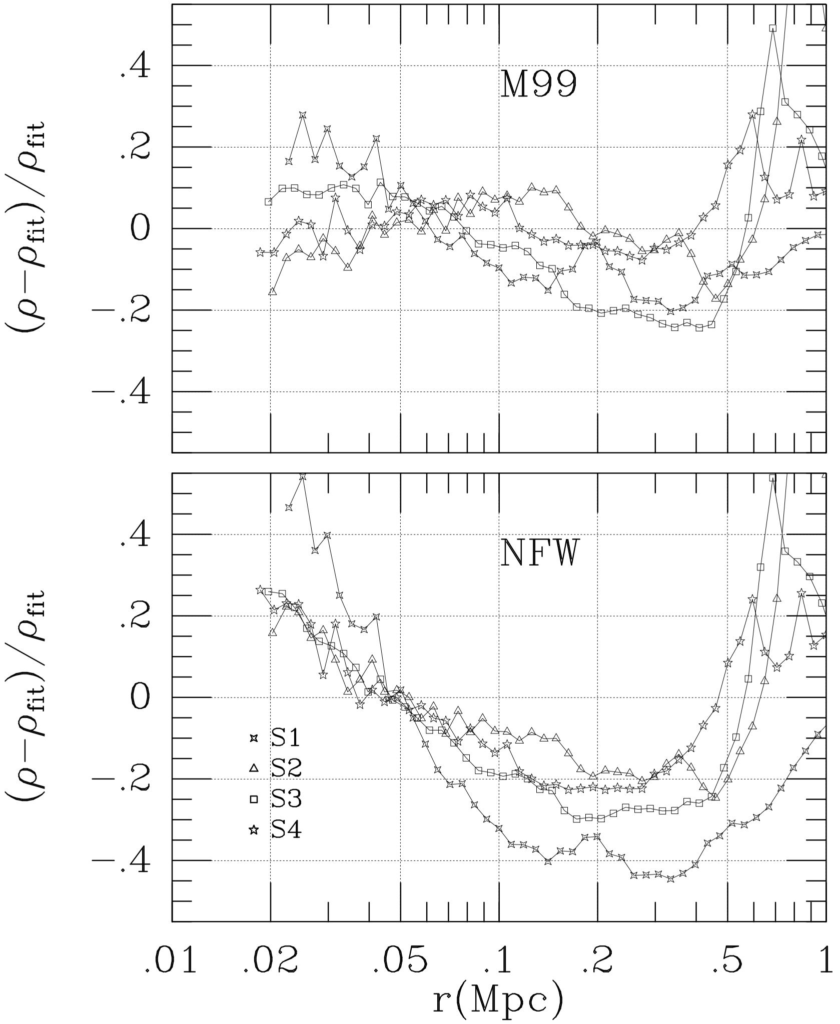
NFW

Moore 99 profile

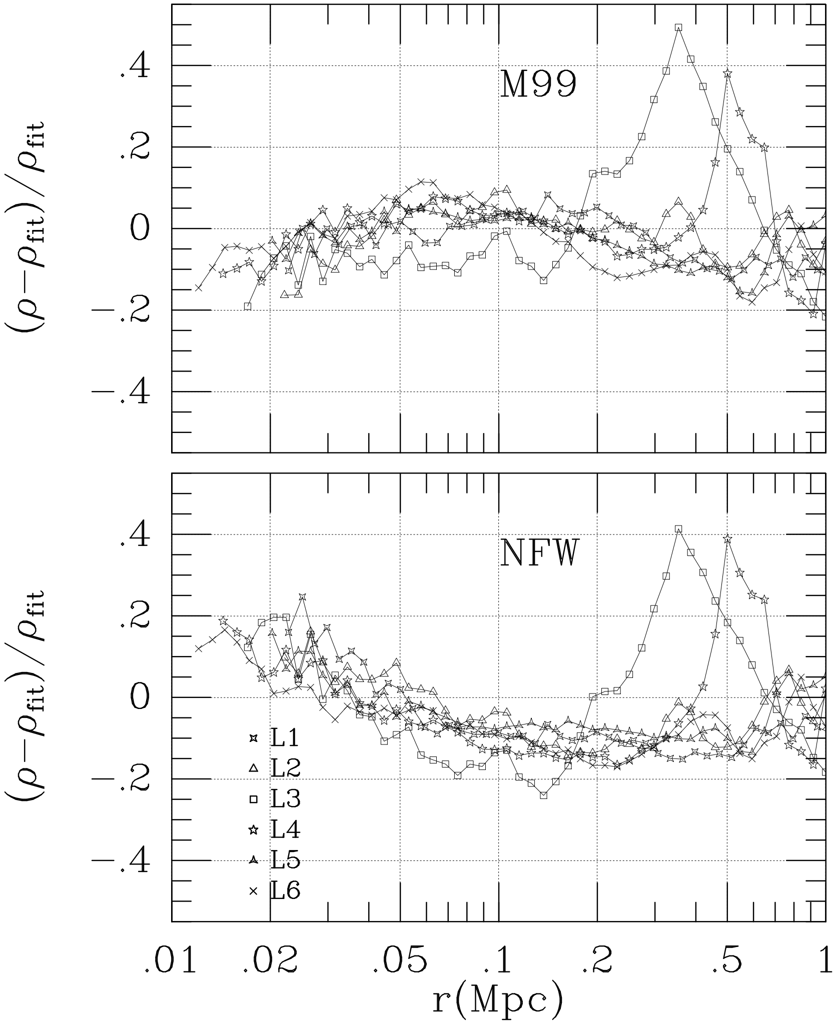
$$\boldsymbol{\rho = \frac{\rho_0}{(r/r_{\mathrm{s}})^{1.5}[1+(r/r_{\mathrm{s}})^{1.5}]}.} \tag{9}$$

との差をくらべてみる

SCDM



LCDM



明らかに **NFW** は悪い

教訓

- 良く知られた仕事だからといって正しいとは限らない
- 人の数値計算が間違っているというためには、人より正しい計算をするのが速い
  - 理屈からは、「正しくないかもしれない」という以上は言えない(というか、本当はいえるんだけど納得してもらうのは難しい)
- 無衝突系なら沢山粒子を入れることでより正しくできる

数値計算の方法等

この種の問題：基本的に

- より大粒子数で
- より正確な

計算をすることで、「新しいことがわかる」(こともある)。

↓

どうやって今までより「良い(大きく、正確な)」計算ができるようにするかが問題

「良い」計算をする方法

- 計算法を改良する
- 速い計算機を買う
- 速い計算機を作る

以下、それぞれの話をする。

計算法

原理的には、多体シミュレーションはとっても単純：

運動方程式

$$\frac{d^2\boldsymbol{x}_i}{dt^2} = \sum_{j \neq i} Gm_j \frac{\boldsymbol{x}_j - \boldsymbol{x}_i}{|\boldsymbol{x}_j - \boldsymbol{x}_i|^3}, \tag{10}$$

を数値積分するだけ。

右辺を計算するプログラム： **2** 重ループで **10** 行くらい

時間積分： なにかルンゲクッタとか適当なものを使えばいい

というだけで話が済めばいいけれど、もちろん世の中はそんなに簡単ではない。

何が問題か？

- 計算精度の問題： 2 粒子の近接散乱、自己重力による構造形成 — 時間刻みをどんどん短くしないとちゃんと計算できなくなる。
  積分時間が長いので高精度の公式を使いたい。
- 計算量の問題： 右辺の計算量が $O(N^2)$ — N が少し大きくなるとすぐに計算時間が現実的ではなくなる

というわけで、以下、時間領域での計算法と空間領域のそれを概観する。

計算法 —— 時間領域

- 独立時間刻み

他に重要な話が沢山あるけど全部省略

独立時間刻み

重力多体問題: 原理的には単に大規模な常微分方程式の初期値問題

適当な公式で時間積分すればいい？

それでは済まない理由:

- 粒子によって非常に大きく軌道のタイムスケールが違うことがある
- 連星とかそういったものができる

構造形成の効果: 最悪 $O(N)$

2 体衝突の効果: $O(N^{1/3})$ 程度

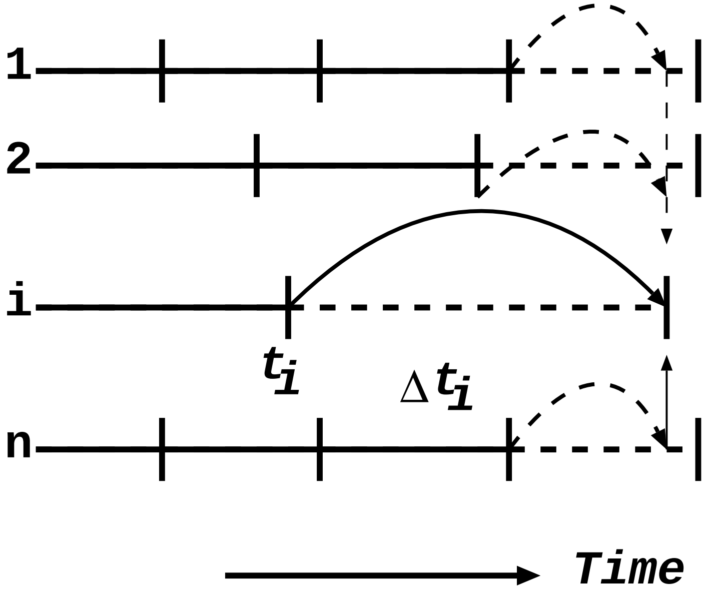
対応

- 粒子毎に時間刻みをバラバラに変化させる。(独立時間刻み)
- 2 体衝突、連星は座標変換して扱う。

独立時間刻みの原理

粒子毎にばらばらの時刻 t_i と時間ステップ $\boldsymbol{\Delta t_i}$ を与える

1. $t_i + \boldsymbol{\Delta} t_i$ が最小の粒子を選ぶ。
2. その粒子の軌道を新しい時刻まで積分する。
3. その粒子の新しい時間刻みを決める。
4. ステップ 1 に戻る。



問題:ある粒子の時刻 $t_i + \boldsymbol{\Delta} t_i$ で他の粒子の位置が必要

他の粒子の位置の計算

時間刻み可変の予測子を使えば問題ない

つまり、

- 各粒子の時間積分公式としては、可変ステップの線形多段階法を PEC モードで使う。
- ある粒子の新しい時刻での加速度を計算するには、他の粒子のその時刻での位置を予測子を使って予測する。

ということになる。

計算法 — 空間領域

運動方程式の右辺をどうやって評価するか？という問題。

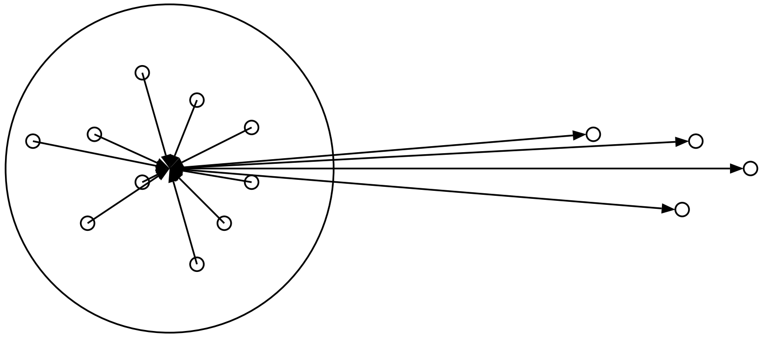
以下、 独立時間刻みのことはとりあえず棚上げにして話をすすめる。

広く使われている方法： Barnes-Hut treecode

有名な方法： 高速多重極法

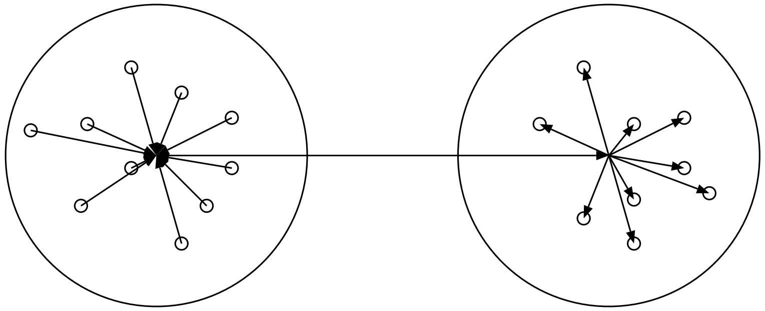
ツリー法、FMMの基本的発想

遠くの粒子
からの力は
弱い


Tree

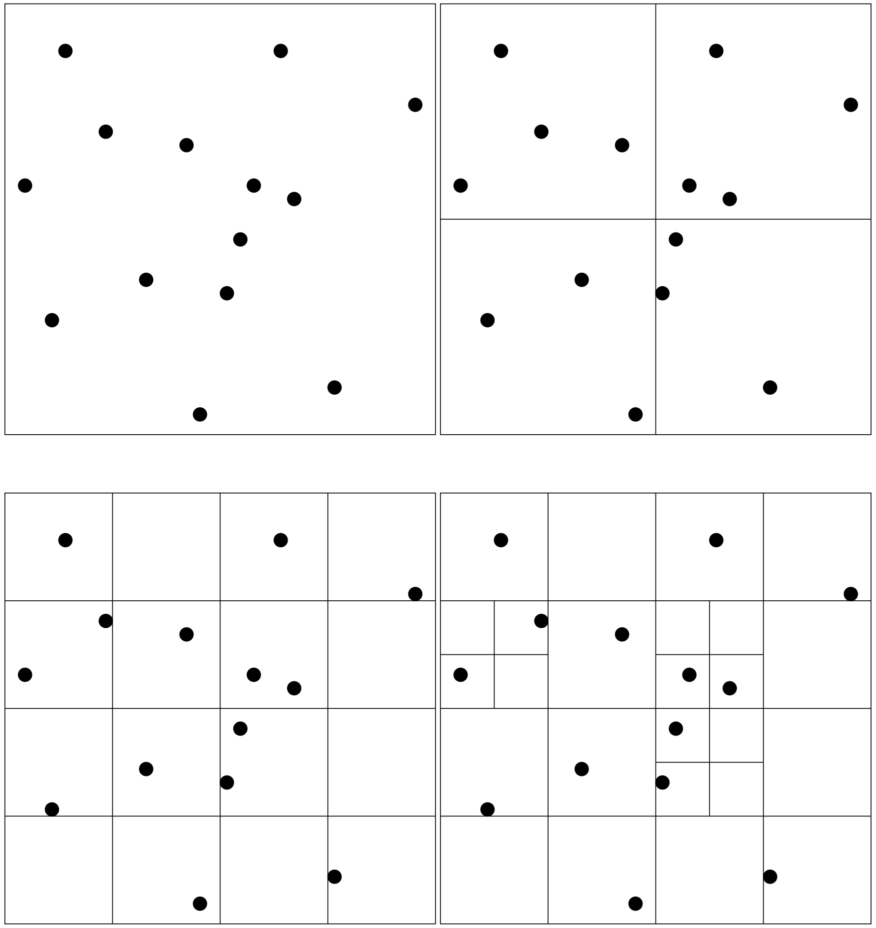
↓

まとめて計
算できな
いか？


FMM

- ツリー：力を及ぼすほうだけをまとめて評価
- **FMM**：力を受けるほうもまとめて評価

どうやってまとめるか? — ツリー法の場合

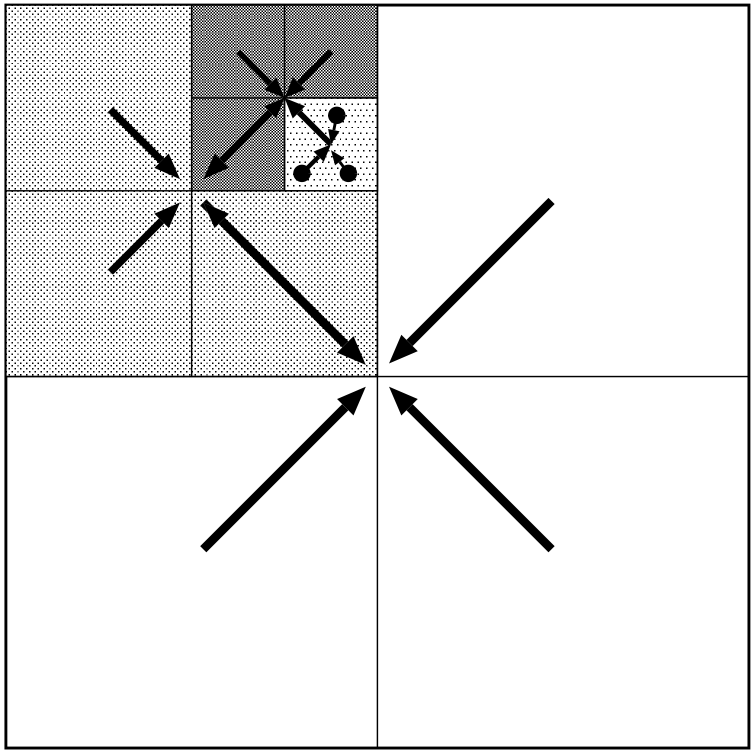


階層的なツリー構造を使う。

- まず、全体が入るセルを作る
- それを再帰的に 8 (2 次元なら 4) 分割する
- 中の粒子がある数以下になったら止める (上の例では 1 個)

多重極展開の構成

まず、ツリーの各セルのなかの粒子がつくるポテンシャルの多重極展開を計算する。



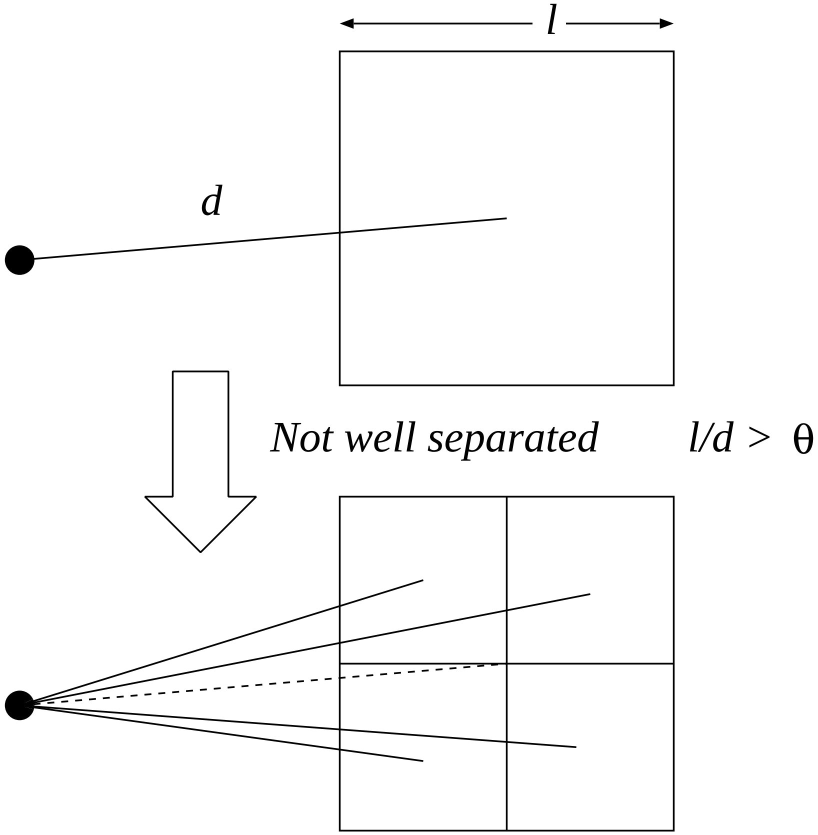
- 最下層のセル:そのなかの粒子が作るポテンシャルを多重極展開
- それ以外 : 子セルの多重極展開の展開中心をシフトして加算

下から順に計算していけばよい。

計算量は $O(N)$。展開をシフトする式はかなり複雑。

ツリー法での力の計算

再帰的な形に表現すると格好がいい。


l
d
Not well separated
l/d > θ

- 十分に離れている:重心(あるいは多重極展開)からの力
- そうでない:子ノードからの力の合計

系全体からの力 ＝ ルートからの力

ツリー法の計算量と計算精度

簡単なオーダー見積り:

$$\text{誤差} \propto \boldsymbol{\theta}^{(\boldsymbol{p}+1)}$$

$$\text{計算量} \propto \boldsymbol{\theta}^{-3}\boldsymbol{p}^{2}\boldsymbol{N}\log\boldsymbol{N}$$

$\boldsymbol{p}$:多重極展開の次数

$\log \boldsymbol{N}$: ツリーの階層数

$\boldsymbol{\theta}^{3}$ ある階層で、相互作用を計算するセルの数

実際には、、、

実際の振舞い：複雑。

- 精度は上の見積りよりよい (θ のべきが大きい)
- 計算量はそれほど増えない

ことが多い。(粒子の分布による)

高速多重極展開法 (FMM)

(今日は省略)

アンダーソンの方法とPseudoparticle method

FMM、高精度ツリーの「実用上の欠点」

- 数学が結構ややこしい：実装、デバッグが大変、高速化も面倒
  (球面調和関数の計算：漸化式なので速くするのが難しい)

もう少し簡単に書けないか？

→ **FMM without Multipoles**

- アンダーソンの方法 (Anderson 1992)
- 疑似粒子法 (Makino 1998)

アンダーソンの方法

多重極展開を球面上でのポテンシャルの値で表す。

球面を境界条件とするラプラス方程式 → ポアソンの公式

ポアソンの公式 (外部解と内部解)

$$\Psi(x) = \frac{1}{4\pi}\int_S \left[\sum_{n=0}^{\infty}(2n+1)\left(\frac{a}{r}\right)^{n+1}\mathrm{P}_n(s\cdot x/|x|)\right]\Psi(as)ds,$$

$$\Psi(x) = \frac{1}{4\pi}\int_S \left[\sum_{n=0}^{\infty}(2n+1)\left(\frac{r}{a}\right)^{n}\mathrm{P}_n(s\cdot x/|x|)\right]\Psi(as)ds,$$

積分を球面上の適当な標本点をつかった数値積分に置き換える

疑似粒子法

基本的な考え：

球面調和関数展開：多数の粒子が作るポテンシャル場の近似

↓

逆に球面調和関数展開を粒子分布で表現できないか?

こういうことを考える理由

- 意味がわかりやすい
- 多体問題専用計算機 (GRAPE) が有効に使える

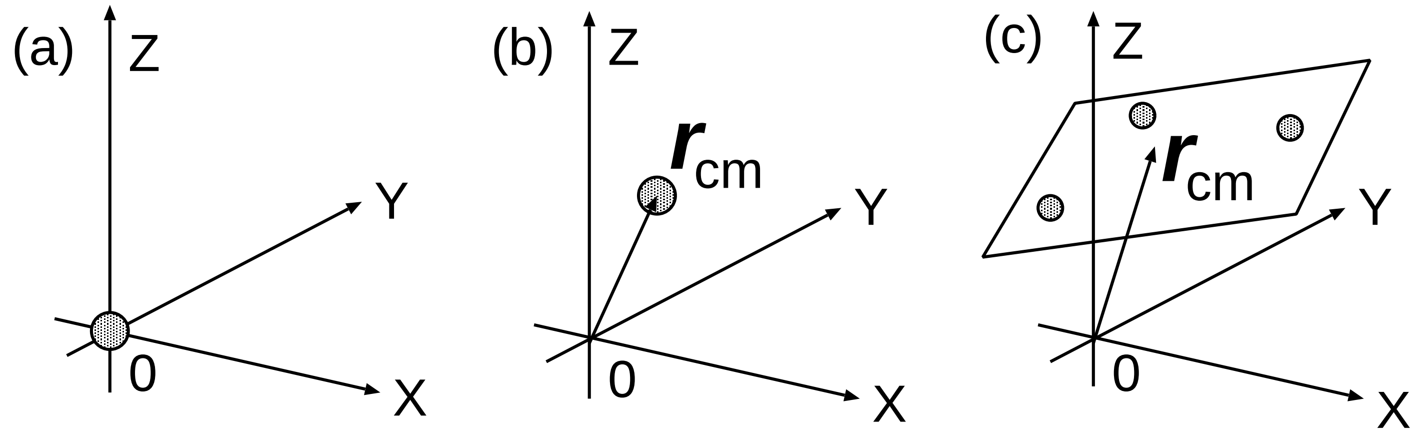
問題：どうやって配置を決めるか？

最小粒子数での実現

$\boldsymbol{p = 0, 1, 2}$ では可能。

$\boldsymbol{p = 1}$**:** 単に重心

$\boldsymbol{p = 2}$ **(Kawai and Makino 1999)**


(a)
Z
Y
0
X
(b)
Z
$\boldsymbol{r}_{\mathrm{cm}}$
Y
0
X
(c)
Z
$\boldsymbol{r}_{\mathrm{cm}}$
Y
0
X

以下では、粒子を置く座標はあらかじめ決めておくやりかたについて考える。

もっと高次 (M2M)

疑似粒子のおき方：アンダーソンの方法と同じく球面上とする。最終的な式：

$$\boldsymbol{m_j = \frac{2l+1}{K}\sum_{i=1}^{N} m_i \sum_{l=0}^{p} (r_i/r)^l P_l(\cos\gamma)}$$

ここで γ は粒子の位置と疑似粒子の位置の間の角度。

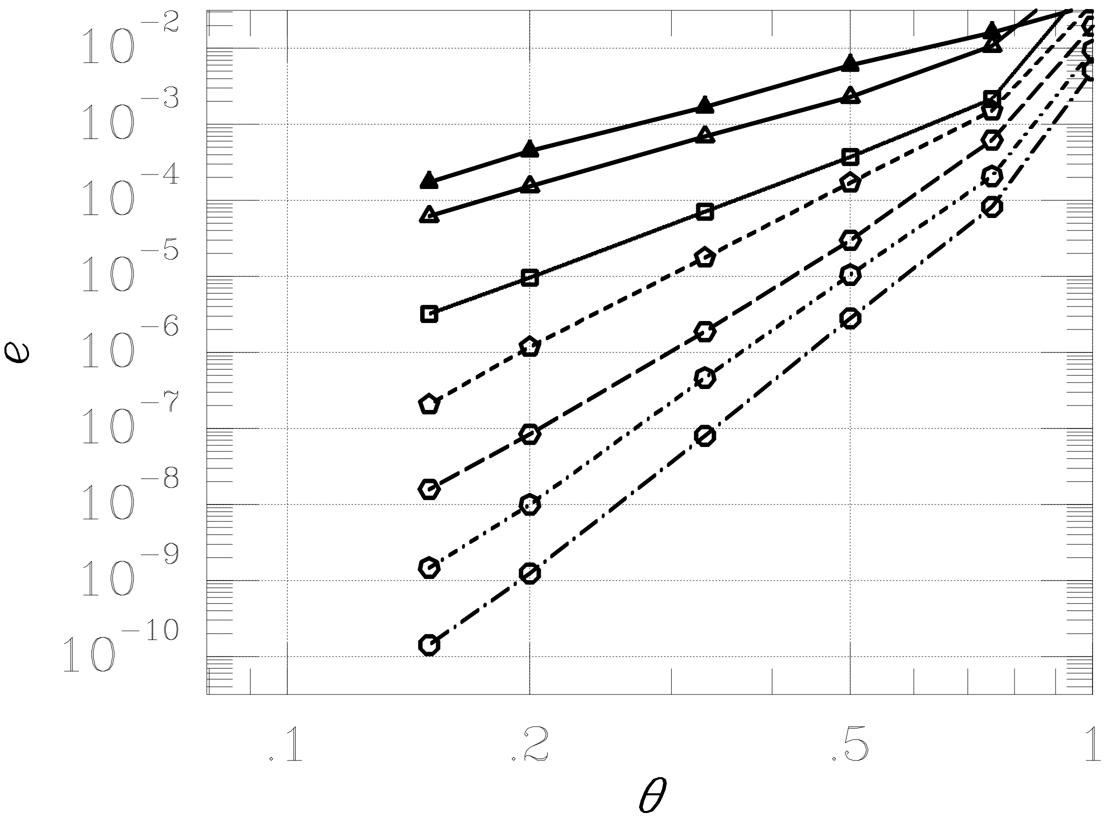
- 上の式は、実粒子から仮想粒子への変換と仮想粒子から上位セルの仮想粒子への変換の 両方に使える。
- 収束性を考えると、球の半径をセルの大きさより 少し小さい程度にとっておくのが良い。

ツリー法との組合せ

計算精度の例

粒子分布は球内一様、

26 万体。



次数は上の **2** つが一次

(定式化が違う) その後

順に **2–6** 次。

ツリー / FMMの効率化・高速化

2方向のアプローチ

- 実装上の効率化
  - ベクトル化、並列化

  この話は後でまとめて。
- アルゴリズムの改良による計算量の減少
  - **FFT**, 回転等

アルゴリズムの「改良」

ツリー法は通常 4 重極まで $\longrightarrow$ アルゴリズムをいじる余地少ない。

天文ではもっと高精度が必要な場合には依然直接計算が主流。(独立時間刻みとの組み合わせが困難なため)

$\mathbf{FMM}$： 多数提案がある。

ほとんどのものが、なんらかの方法で $\mathbf{M2L}$ の計算量のオーダーを下げようというもの(こことと近くの直接計算の計算量が主要)

役に立つかどうかは???

独立時間刻みとツリー、**FMM**

- 組み合わせることは可能 **(McMillan and Aarseth 1993)**
- 高精度なものの **MPP** への実装は存在しない
- **GRAPE** への実装もない

実装が困難な根本的な理由：計算量が減った分 並列度が下がっているため

今後の大きな課題といえる。

高速化の第二の方法: 速い計算機を使う

速い計算機を使えば速く計算できるというのはなんかトートロジーみたいで当たり前という気もするが、実はそうでもない。

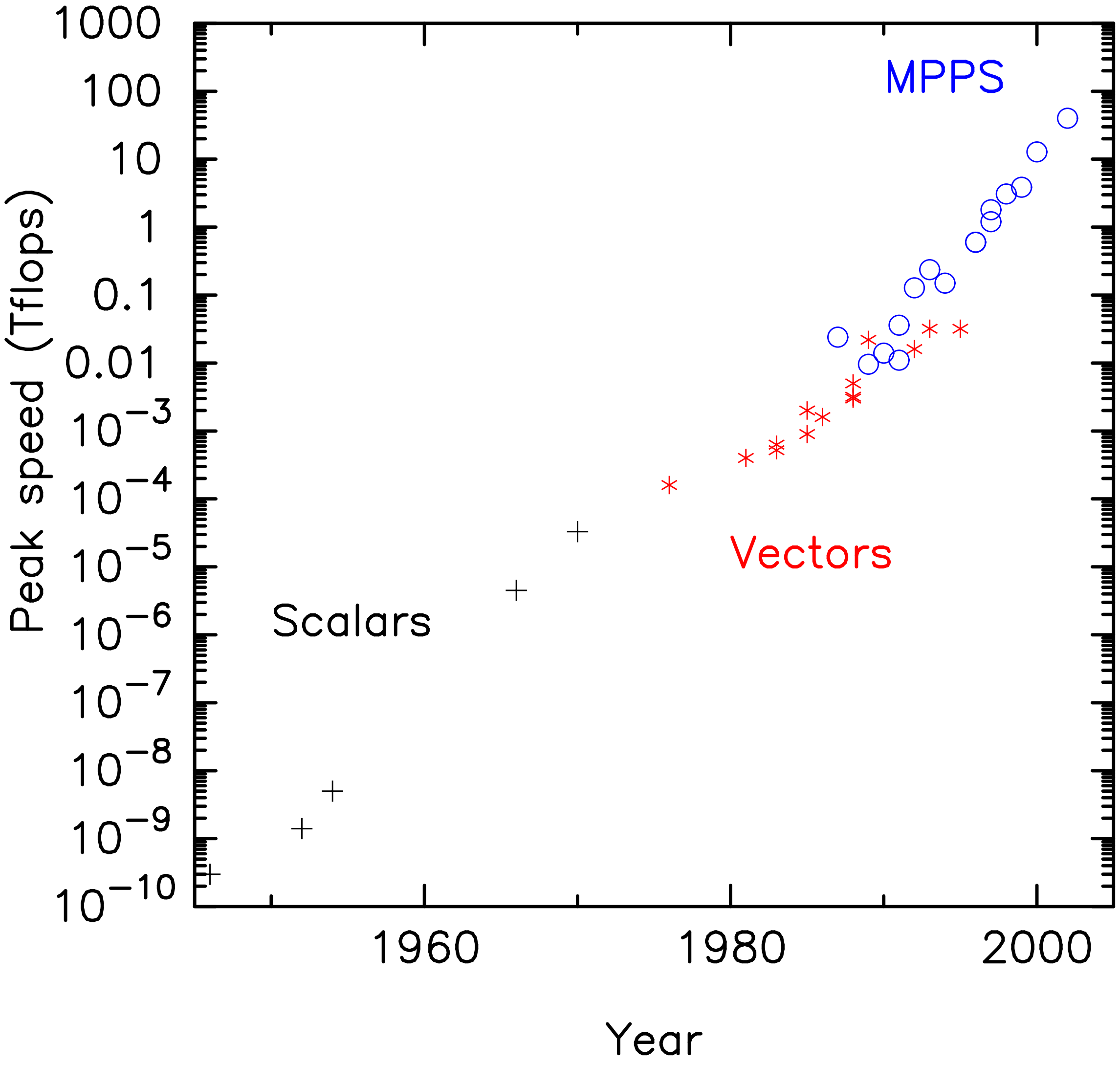
根本的な理由:

最近 30 年間の計算機の「発展」のおかげで、あるアルゴリズムを速い計算機でちゃんと性能がでるように動かすのはどんどん大変になってきた。

計算機の発展

速度の進化: **50** 年で $\mathbf{10^{10}}$ 倍

ほぼ時間の指数関数



並列計算機

並列計算機といっても、共有メモリならベクトル計算機と同じ問題が発生 — **Memory wall**

共有メモリにしない → 問題が (とりあえず) 回避される。

つまり: **1 CPU + メモリ**の簡単な計算機を構成要素にして、それを沢山並べる。

その間は、なんか適当なネットワークとかでつながる

という方向だと、ややこしい問題を (計算機を作る人は) 考えなくてもいい。

＝分散メモリ型並列計算機

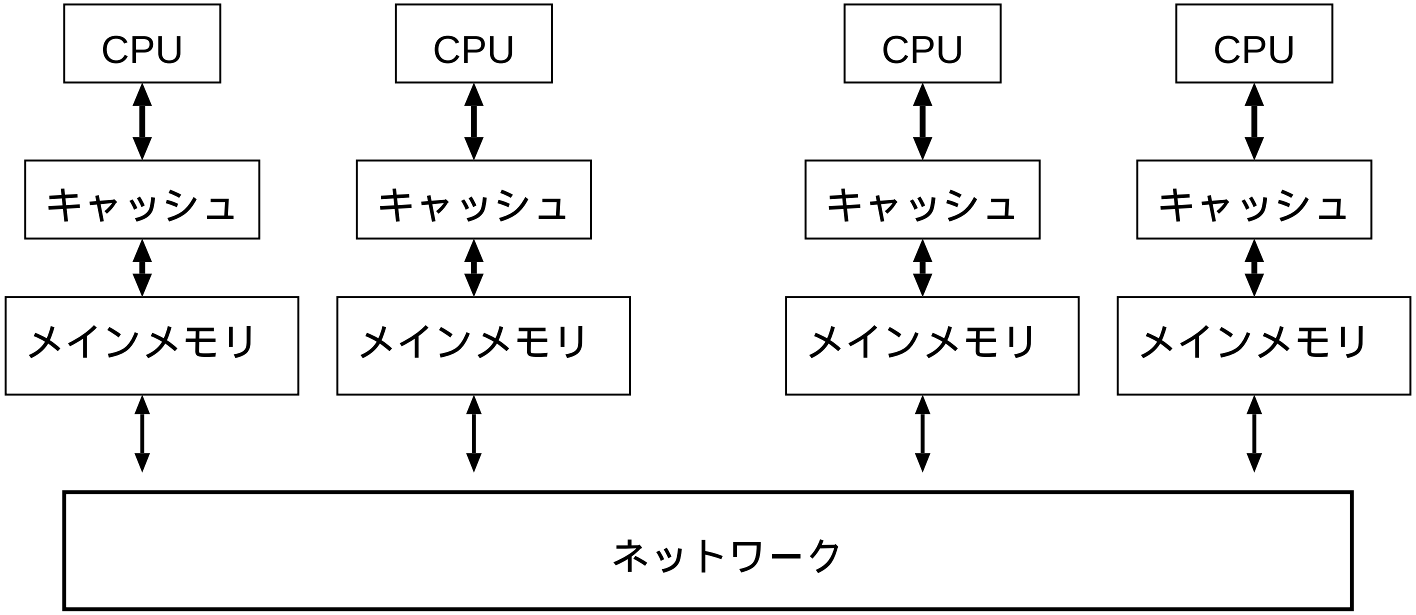
分散メモリ型並列計算機

開発の重要な源流は

- QCD 計算のための計算機 (**Caltech Hypercube**)
- 星野-川合らによる **PACS/PAX**
- イリノイ大学の **ILLIAC IV**

どれも、割合単純な連続系 (偏微分方程式系) を解くことだけを目標に設計・製作された。

分散メモリ型計算機の構成


CPU
CPU
CPU
CPU
キャッシュ
キャッシュ
キャッシュ
キャッシュ
メインメモリ
メインメモリ
メインメモリ
メインメモリ
ネットワーク

どれもおなじようなもの。**ILLIAC IV** は特殊: **SIMD** 型 (全部の **CPU** が同じ演算を実行)。

あとの流れは **MIMD** (みんなバラバラに計算する)

分散メモリ型の中でも特に台数が多いもの: **Massively-Parallel** とか **Highly-Parallel** と呼んでいた。

分散メモリ型計算機の歴史

1970年代: 汎用マイクロプロセッサを使った計算ノード

1980年代: 多様

- 汎用マイクロプロセッサ/専用プロセッサ
- SIMD/MIMD
- 分散メモリ/分散共有メモリ/AllCache/...

分散メモリ型計算機の歴史 (2)

1990 年代: 淘汰

- ベクトル型分散メモリ— 富士通、**NEC**
- **Cray T3x — Alpha**
- **Beowulf — Intel x86**

2000 年代: ???

- **Beowulf**、**SMP** クラスタ以外は死滅**?**

死滅した理由

- 価格性能比
- プログラミングモデル

価格性能比

消費者マーケットに出るもの: 生産数が何桁も違う。
それだけ、開発費を投入して量産コストを下げることができる。

2002年時点で:

生産数 $< 10^4$ 個のカスタム **LSI:** 開発費 数億、量産コストチップあたり **5-10** 万動作クロック **300-500 MHz**

Intel Pentium 4: 開発費 ??? 売値: 安いものなら **1** 万、動作クロック: **2GHz** 以上

同じようなものを作ると、価格性能比で **10** 倍は損

プログラミングモデル

多様なアーキテクチャ → プログラミングモデルへの要求:

- 移植性
- 性能の高さ

生き残ったモデル: **message-passing**

要するに: 各ノードでは普通にプログラムが走る。必要に応じて、通信ライブラリを使って通信する。

自動並列化とか並列処理言語とかそういったものには頼らない。

もっとも安直なハードウェアに有利な状況

Beowulf 上の並列プログラム

境界条件

- メッセージパッシングによるプログラム
- ネットワークを通したデータ転送
  - レイテンシは(特殊なハード、ソフトを使わないと) 遅い。 $> 100\mu$s
    「遅い」主記憶に比べてもさらに3桁遅い
  - バンド幅はどうやっても細い。今なら **GbE** で **100MB/s**
    主記憶に比べてもさらに数十倍遅い

アルゴリズムに要求される条件

通信回数、通信量が十分に少ないこと。

多体シミュレーションの並列化

- 独立時間刻み
- ツリー・FMM

独立時間刻みの並列化

実はそれ以前の問題: 「全粒子から全粒子への力の計算」の並列化

並列化の 2 方法

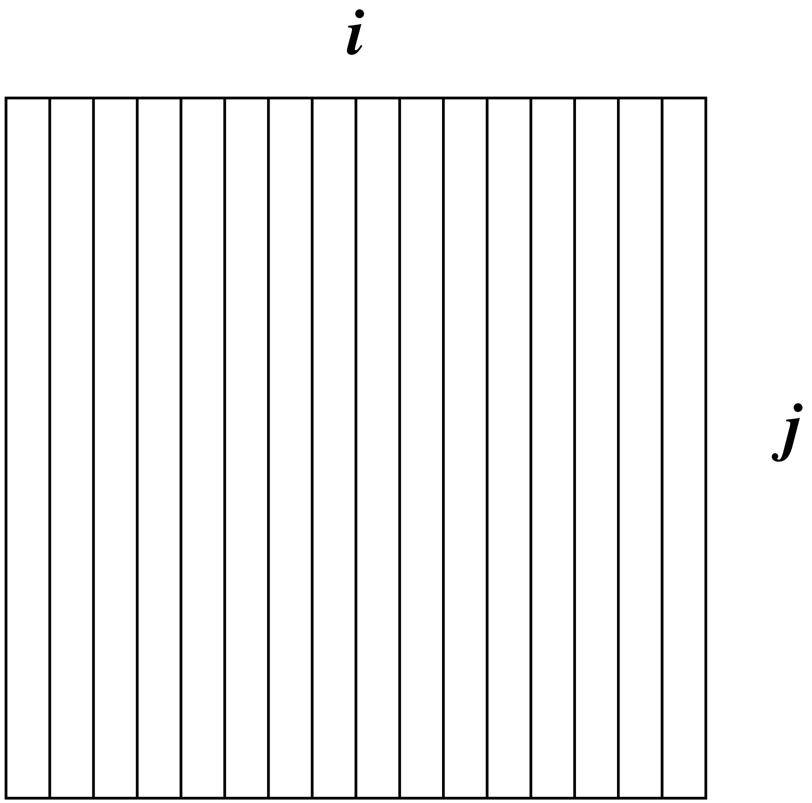
- 各ノードが全粒子のコピーを持って、重力計算、時間積分は分担する
- 各ノードが自分の分担の分だけを持つ。重力計算のためには他のノードから粒子を貰ってくる。

どちらにしても、各ノードが全粒子の情報を使って力を計算する。

ノード一つの通信バンド幅でスケーラビリティが制限される。

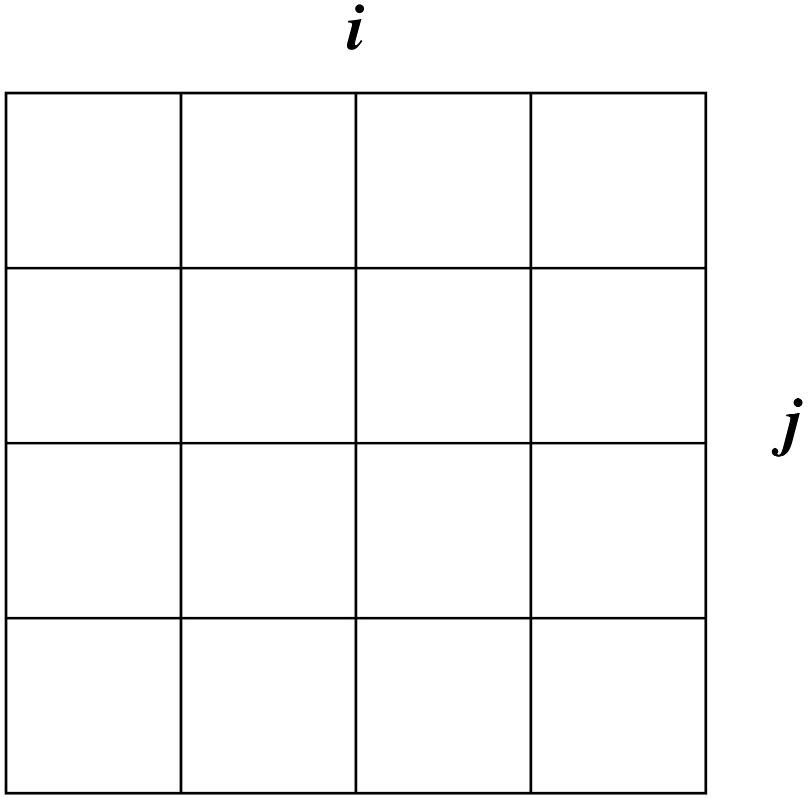
並列化しても速くならない。

遠距離相互作用 —— 相互作用行列を考える



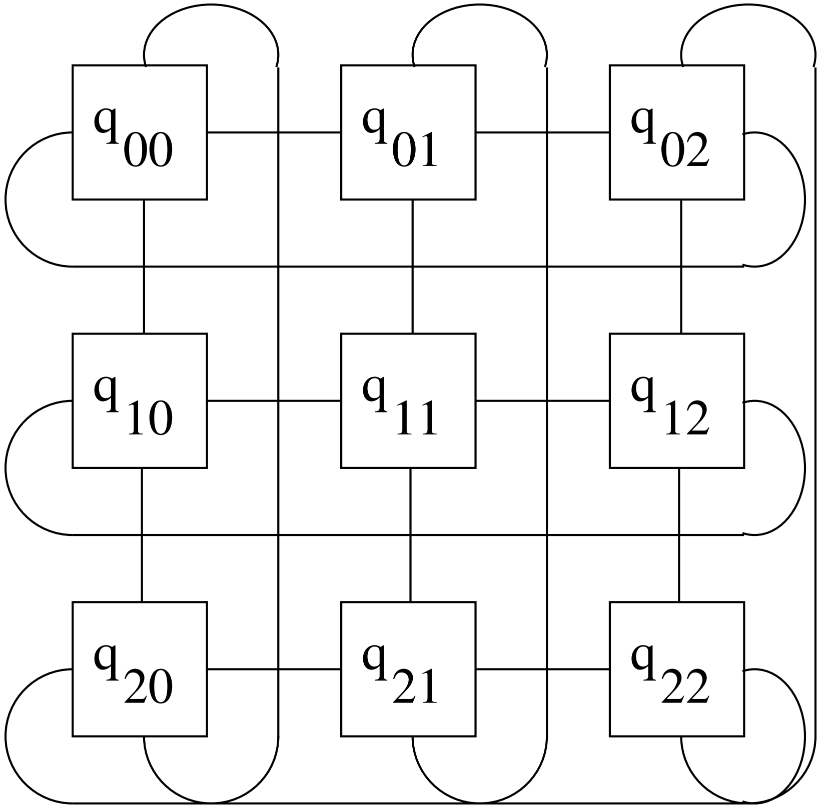
- 相互作用の行列を見ると、普通のアルゴリズムは1次元分割になっている。
- これがスケーラブルでない理由
- 2次元分割は可能か？

相互作用の2次元分割



- 形式的には左のような絵を書ける。
- 各小行列にプロセッサを割り当てる
- 独立時間刻みにも応用できる。

ハードウェアとしては、、、



q_{00}
q_{01}
q_{02}
q_{10}
q_{11}
q_{12}
q_{20}
q_{21}
q_{22}

- 分割した相互作用行列にプロセッサを対応させる。
- トポロジ：**2D** トーラスなり「ハイパークロスバー」なり、、、

ツリー法のベクトル化、並列化

ベクトル化、並列化の技法はツリー法も FMM も同じ。

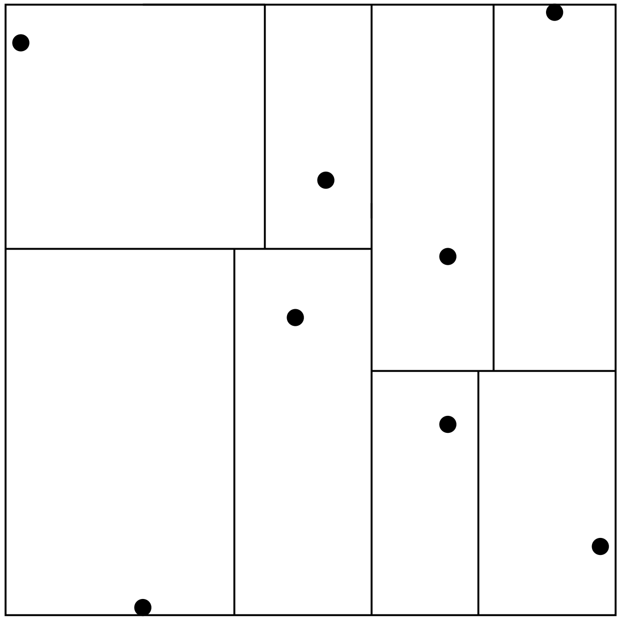
- 共有メモリの機械では容易 (アルゴリズムの並列度は実際上 N)
- メッセージパッシングの機械 : 効率の良い空間分割が必要。**Salmon & Warren** が様々な方法をテストした。プログラムは面倒だが、作ってしまえば効率はよい。

従来の並列化アルゴリズム

実用になっている並列化法は以下の2つ。どちらももと Caltech Hypercube のグループの Salmon と Warren によるもの

- **Orthogonal Recursive Bysection (ORB)**
- **Hashed Oct Tree (HOT)**

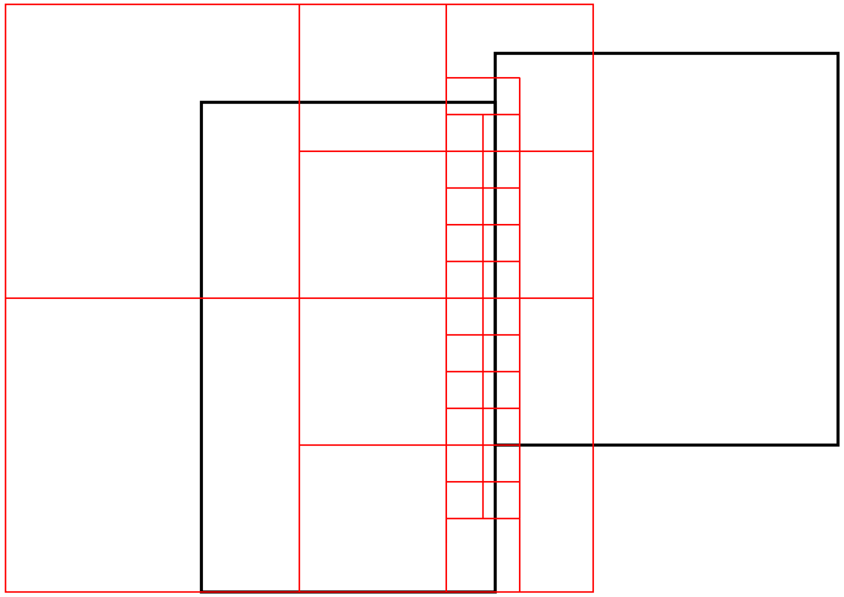
ORB



- まず、粒子が半分ずつにわかれるように x 軸に垂直な平面で切る
- 切ったそれぞれについて、また半分になるように y 軸に垂直な平面で切る 3 次元なら次に z、2 次元なら次は x 軸になる

というのを、プロセッサ数になるまで繰り返す

他のプロセッサにある粒子からの力の計算

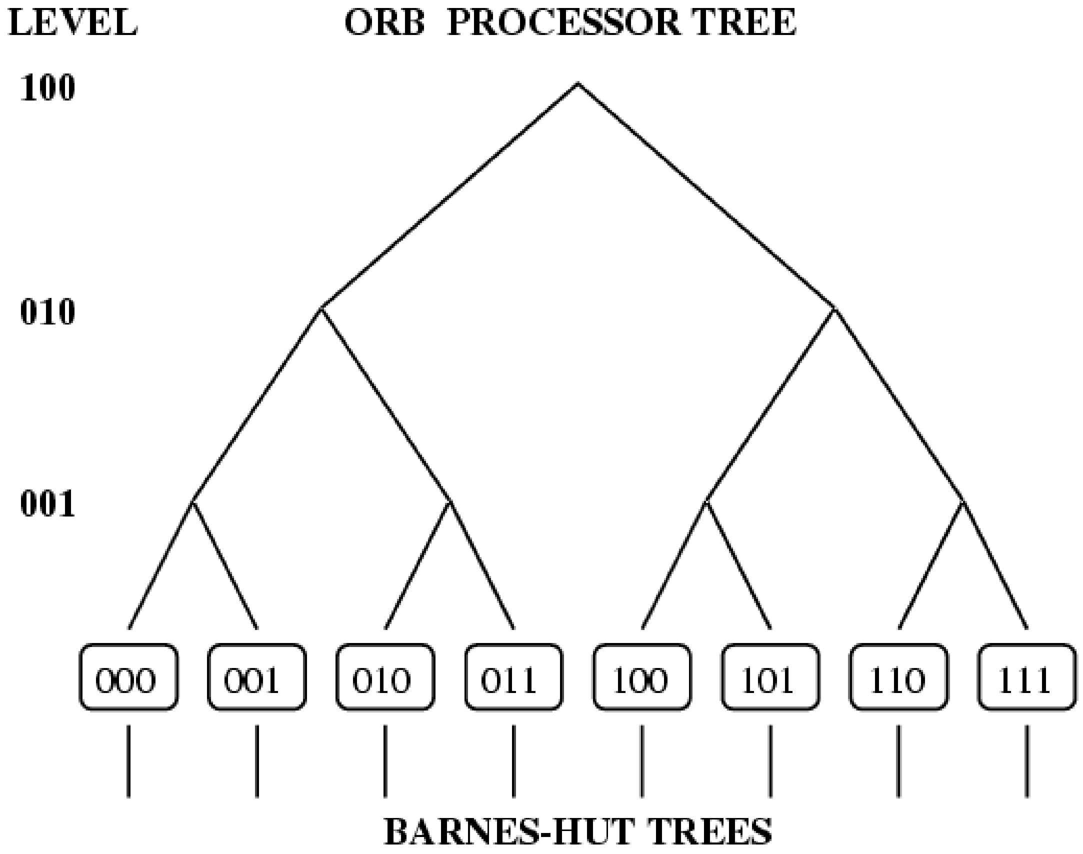


他のプロセッサから、「必要ないところをカットした」ツリーをもらってくる (**local essential tree, LET**)

それらと自分のところを適当にまとめて全体ツリーを作る

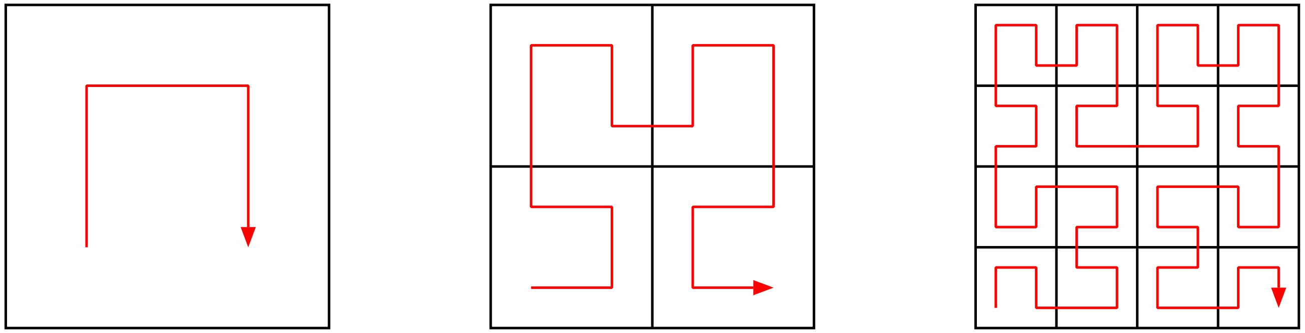
「適当にまとめる」とは？

Dubinski の実装：この上の階層は ORB ツリーそのもの


LEVEL
ORB PROCESSOR TREE
100
010
001
000
001
010
011
100
101
110
111
BARNES-HUT TREES

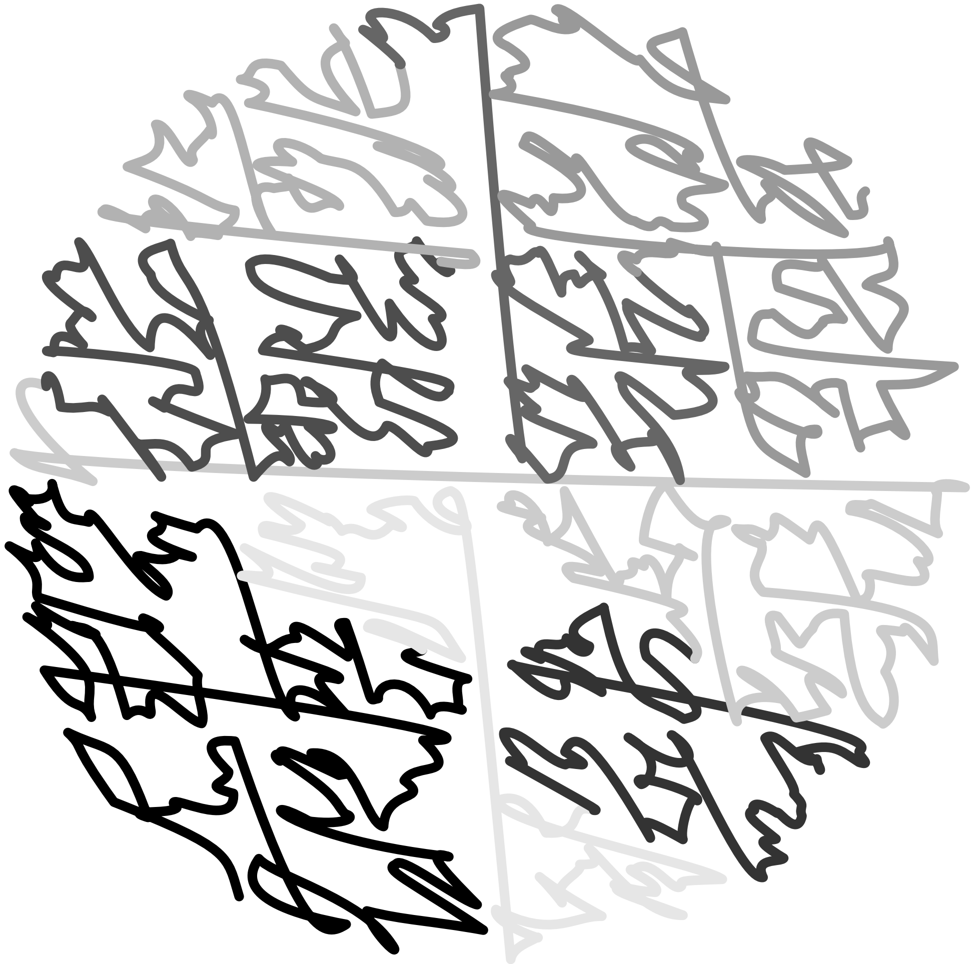
HOT

ペアノ曲線：N 次元空間を 1 次元にマッピング



- 粒子をペアノ曲線上の順番で並べる
- 各プロセッサに連続した粒子を割り当てる

HOT の例



ただしこれはペアノ曲線ではなくて **Morton Ordering**

HOT でのツリー構成・重力計算

ツリー構成

- ペアノ曲線を作るところでツリーが構成される
- 並列ソートが必要

重力計算

- **LET** のようなことは困難
- 計算中に必要になった情報を他のプロセッサにリクエストする。

我々の使うことにした方法

ORB を多少変更 **(Makino 2003)**

- 深さを **3** 層までに制限
- **2** 分割以上も許す

例えば、 **12** ノードなら $\mathbf{3 \times 2 \times 2}$ とかいうふうに分ける。
もちろん、 **2, 4, 8** ノードでは従来の **ORB** と同じ。