Parallel Multipole Methods

Paul Beckman, Mariya Savinov

NYU Courant

May 2, 2022

Motivating Problem

Motivating Problem

- Consider a collection of interacting particles with a potential

Motivating Problem

- Consider a collection of interacting particles with a potential
- **Total potential** at a point x due to **particles** x_j with **charges** q_j is

$$u(x) = \sum_{j=1}^{n} \frac{q_j}{|x - x_j|} = \sum_{j=1}^{n} q_j \phi(x - x_j)$$

Motivating Problem

- Consider a collection of interacting particles with a potential
- **Total potential** at a point x due to **particles** x_j with **charges** q_j is

$$u(x) = \sum_{j=1}^{n} \frac{q_j}{|x - x_j|} = \sum_{j=1}^{n} q_j \phi(x - x_j)$$

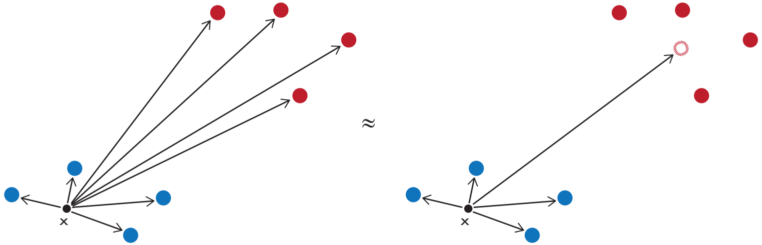
- Separate sum into **near-field** and **far-field** contributions

Motivating Problem

- Consider a collection of interacting particles with a potential
- **Total potential** at a point x due to **particles** x_j with **charges** q_j is

$$u(x) = \sum_{j=1}^{n} \frac{q_j}{|x - x_j|} = \sum_{j=1}^{n} q_j \phi(x - x_j)$$

- Separate sum into **near-field** and **far-field** contributions
  - Take *aggregate* effect of far-field charges via **Taylor series expansion**



Barnes-Hut Tree Code: Taylor Expansion

Barnes-Hut Tree Code: Taylor Expansion

- For a set of points x_j, potential contribution can be approximated:

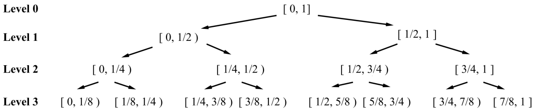
$$\sum_{j \in \text{far-field}} q_j \phi(x - x_j) \approx \sum_{m=0}^{p} \underbrace{\left[\sum_{j \in \text{far-field}} q_j a_m(x_j - x^*) \right]}_{\text{weight } m} S_m(x^* - x) + O(\delta^{p+1})$$

Barnes-Hut Tree Code: Taylor Expansion

- For a set of points x_j, potential contribution can be approximated:

$$\sum_{j \in \text{far-field}} q_j \phi(x - x_j) \approx \sum_{m=0}^{p} \underbrace{\left[\sum_{j \in \text{far-field}} q_j a_m(x_j - x^*) \right]}_{\text{weight } m} S_m(x^* - x) + O(\delta^{p+1})$$

- For N particles, partition $[0, 1]$ uniformly at $O(\log N)$ levels:

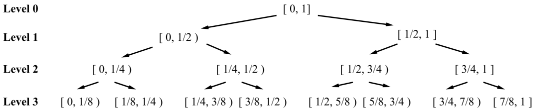


Barnes-Hut Tree Code: Taylor Expansion

- For a set of points x_j, potential contribution can be approximated:

$$\sum_{j \in \text{far-field}} q_j \phi(x - x_j) \approx \sum_{m=0}^{p} \underbrace{\left[\sum_{j \in \text{far-field}} q_j a_m(x_j - x^*) \right]}_{\text{weight } m} S_m(x^* - x) + O(\delta^{p+1})$$

- For N particles, partition $[0,1]$ uniformly at $O(\log N)$ levels:



- For each $T_{\ell,k}$ cell, compute weights $w_{\ell,k,m} \longrightarrow$ total cost $O(N \log N)$

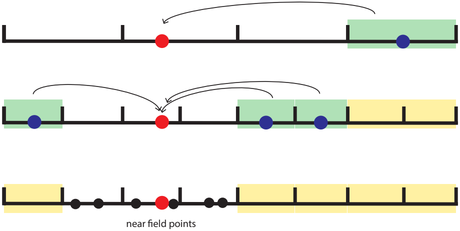
Tree Code by Barnes and Hut: Interaction List

Tree Code by Barnes and Hut: Interaction List

- At each level, only 'interact' with up to 3 far-field cells *not already included at higher levels*

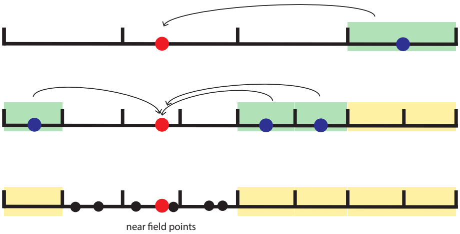
Tree Code by Barnes and Hut: Interaction List

- At each level, only 'interact' with up to 3 far-field cells *not already included at higher levels*



Tree Code by Barnes and Hut: Interaction List

- At each level, only 'interact' with up to 3 far-field cells *not already included at higher levels*

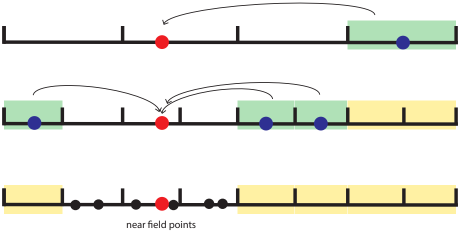


- Then for a point x, the potential is approximately

$$u(x) = \sum_{\ell=1}^{O(\log N)} \sum_{m=0}^{p} w_{\ell,k(\ell),m} S_m \left(x^*_{\ell,k(\ell)} - x \right) + O\left(2^{-p}\right)$$

Tree Code by Barnes and Hut: Interaction List

- At each level, only 'interact' with up to 3 far-field cells *not already included at higher levels*



- Then for a point x, the potential is approximately

$$u(x) = \sum_{\ell=1}^{O(\log N)} \sum_{m=0}^{p} w_{\ell,k(\ell),m} S_m \left(x^*_{\ell,k(\ell)} - x \right) + O\left(2^{-p}\right)$$

- Total cost is $O(N \log N)$

32 points in finest box, p=5

Parallelism

Parallelism

- Parallelizing with OpenMP

Parallelism

- Parallelizing with OpenMP
  - *Challenge:*

Parallelism

- Parallelizing with OpenMP
  - *Challenge:* Recursive tree code may prevent significant speedup

Parallelism

- Parallelizing with OpenMP
  - *Challenge:* Recursive tree code may prevent significant speedup
- Two options of how to parallelize

Parallelism

- Parallelizing with OpenMP
  - *Challenge:* Recursive tree code may prevent significant speedup
- Two options of how to parallelize
- **OPTION 1:** Parallelize the loops in the code

Parallelism

- Parallelizing with OpenMP
  - *Challenge:* Recursive tree code may prevent significant speedup
- Two options of how to parallelize
- **OPTION 1:** Parallelize the loops in the code
  - computing weights, adding near-field terms, and adding far-field terms

Parallelism

- Parallelizing with OpenMP
  - *Challenge:* Recursive tree code may prevent significant speedup
- Two options of how to parallelize
- **OPTION 1:** Parallelize the loops in the code
  - computing weights, adding near-field terms, and adding far-field terms
  - *Potential Problem:* Loops of varying sizes, possibly large overhead

Parallelism

- Parallelizing with OpenMP
  - *Challenge:* Recursive tree code may prevent significant speedup
- Two options of how to parallelize
- **OPTION 1:** Parallelize the loops in the code
  - computing weights, adding near-field terms, and adding far-field terms
  - *Potential Problem:* Loops of varying sizes, possibly large overhead
- **OPTION 2:** Distribute the work in the tree among threads

Parallelism

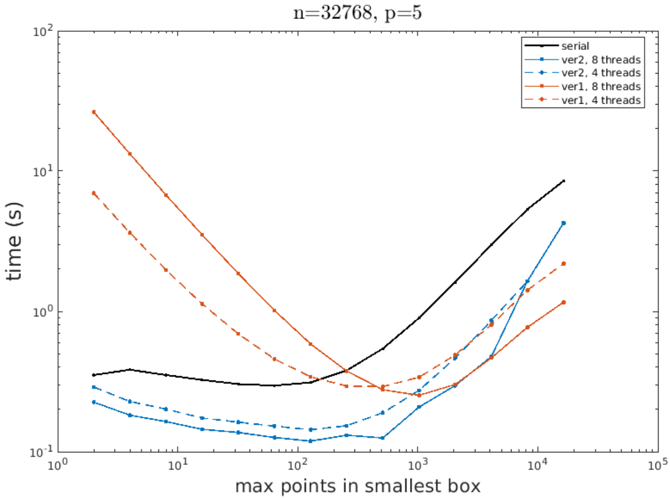
- Parallelizing with OpenMP
  - *Challenge:* Recursive tree code may prevent significant speedup
- Two options of how to parallelize
- **OPTION 1:** Parallelize the loops in the code
  - computing weights, adding near-field terms, and adding far-field terms
  - *Potential Problem:* Loops of varying sizes, possibly large overhead
- **OPTION 2:** Distribute the work in the tree among threads
  - At tree levels $\ell < O(\log r)$ for r threads, open a new section

Parallelism

- Parallelizing with OpenMP
  - *Challenge:* Recursive tree code may prevent significant speedup
- Two options of how to parallelize
- **OPTION 1:** Parallelize the loops in the code
  - computing weights, adding near-field terms, and adding far-field terms
  - *Potential Problem:* Loops of varying sizes, possibly large overhead
- **OPTION 2:** Distribute the work in the tree among threads
  - At tree levels $\ell < O(\log r)$ for r threads, open a new section
  - *Potential Problem:* Nested parallelism

Results: Comparing Parallel Options

Results: Comparing Parallel Options

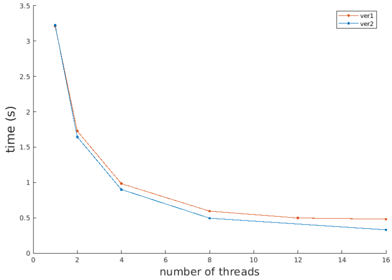


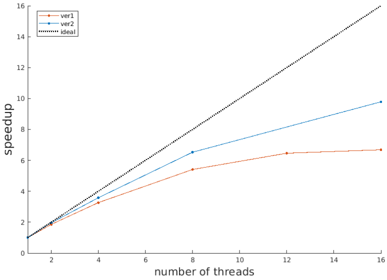
`crackle1`: Intel Xeon E5630 (2.53 GHz)

Results: Strong Scaling

Results: Strong Scaling

$n = 32768$, max points in box 2048, and $p = 5$



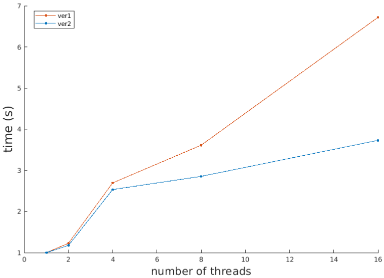


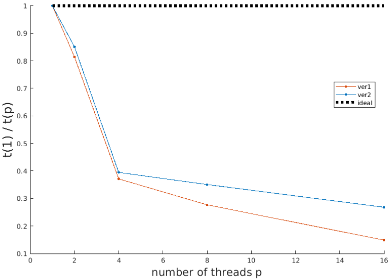
`crunchy1`: AMD Opteron 6272 (2.1 GHz)

Results: Weak Scaling

Results: Weak Scaling

$n = 8192 \cdot r$, max points in box $= 2048$, and $p = 5$





`crunchy1`: AMD Opteron 6272 (2.1 GHz)

Supplemental Slides

Approximating far-field

- Use **Taylor series expansion** of $\phi(x - x_j)$ for small $\delta = \dfrac{x_j - x^*}{x^* - x}$:

$$\begin{aligned}
\phi(x_j - x) &= \phi\left((x^* - x)(1 + \delta)\right) = \phi(x^* - x)\phi(1 + \delta) \\
&\approx \phi(x^* - x)\left[\sum_{m=0}^{p} \frac{\phi^{(m)}(1)}{m!}\delta^m + O(\delta^{p+1})\right] \\
&= \sum_{m=0}^{p} a_m(x_j - x^*)S_m(x^* - x) + O(\delta^{p+1})
\end{aligned}$$

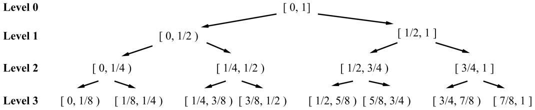
- Then potential from $x_j \in$ far-field is

$$\sum_{j \in \text{far-field}} q_j \phi(x - x_j) \approx \sum_{m=0}^{p}\left[\sum_{j \in \text{far-field}} q_j a_m(x_j - x^*)\right] S_m(x^* - x) + O(\delta^{p+1})$$

- Accuracy depends on choice of x^* and thus size of $\delta = \dfrac{x_j - x^*}{x^* - x}$

Tree Algorithm by Barnes and Hut, complexity $O(N \log N)$

- For N particles, partition $[0, 1]$ uniformly at $O(\log N)$ levels:



- Let $T_{\ell,k}$ be the cell at level ℓ with index $k = 1 : 2^\ell$ with center $x^*_{\ell,k}$.
- Compute weight at each cell, total cost $O(N \log N)$

$$w_{\ell,k,m} = \sum_{x_j \in T_{\ell,k}} q_j a_m(x_j - x^*_{\ell,k})$$

- For a point x: far-field components added at increasingly *coarse* levels

$$u(x) = \sum_{\ell=1}^{O(\log N)} \sum_{m=0}^{p} w_{\ell,k(\ell),m} S_m \left(x^*_{\ell,k(\ell)} - x \right)$$