# The Fast Multipole Algorithm vs. the Particle Mesh Ewald Method

Joshua Nelson, u4850020

COMP3006 - Research Project

November 1, 2012

# The N body problem
## Statement of the problem



*Given N bodies that all interact with each other in some way, how do we efficiently calculate the effect each body has on every other body?*

# The N body problem
## History

- First formally specified by Isaac Newton

# The N body problem
## History



- First formally specified by Isaac Newton
- No exact formula solution

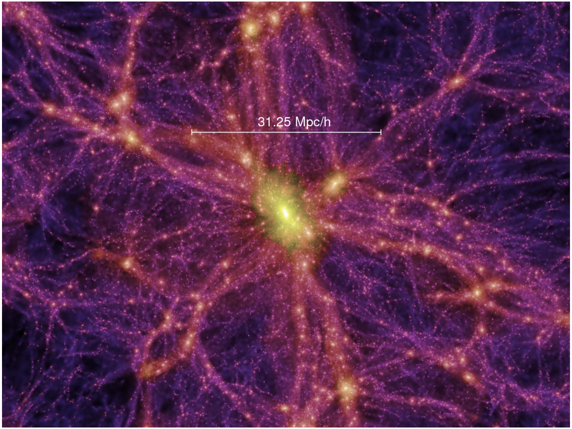
# The N body problem

## History

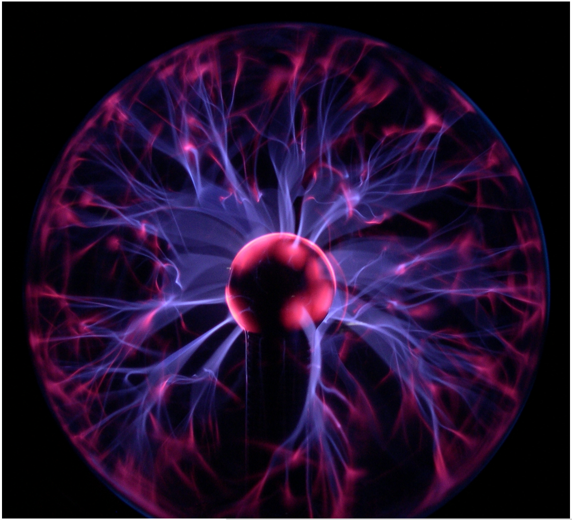
- First formally specified by Isaac Newton
- No exact formula solution
- Approximation methods devised

# The N body problem

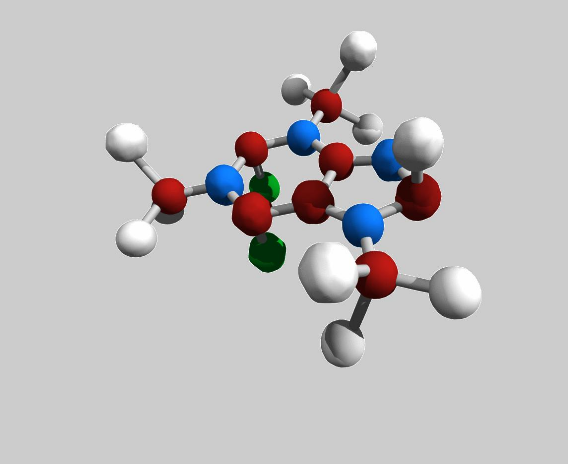
## Applications



# The N body problem
## Applications



# The N body problem
## Applications



# The basic solution
## Algorithm

- Say that a particle at position $r_i$ with charge $q_i$ gives a potential $Q$ at a particle $r_j$ with charge $q_j$ according to the formula

$$Q = \frac{q_i * q_j}{|r_i - r_j|}$$

# The basic solution
## Algorithm

- Say that a particle at position $r_i$ with charge $q_i$ gives a potential $Q$ at a particle $r_j$ with charge $q_j$ according to the formula

$$Q = \frac{q_i * q_j}{|r_i - r_j|}$$

- For each particle pair, calculate the the interaction according to the above formula

# The basic solution
## Algorithm



- Say that a particle at position $r_i$ with charge $q_i$ gives a potential $Q$ at a particle $r_j$ with charge $q_j$ according to the formula

$$Q = \frac{q_i * q_j}{|r_i - r_j|}$$

- For each particle pair, calculate the the interaction according to the above formula
- Use the potential at each particle to calculate the force, and move the particle according to this force.

# The basic solution

## Algorithm

- Say that a particle at position $r_i$ with charge $q_i$ gives a potential $Q$ at a particle $r_j$ with charge $q_j$ according to the formula

$$Q = \frac{q_i * q_j}{|r_i - r_j|}$$

- For each particle pair, calculate the the interaction according to the above formula
- Use the potential at each particle to calculate the force, and move the particle according to this force.
- Order $O(n^2)$ complexity.

# The Fast Multipole Algorithm

## Key components to the Fast Multipole Algorithm

Key components to the Fast Multipole Algorithm

- The Mesh

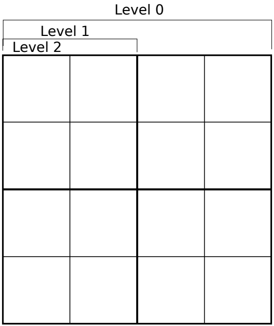
Key components to the Fast Multipole Algorithm

- The Mesh
- Multipole Expansions

# The Fast Multipole Algorithm

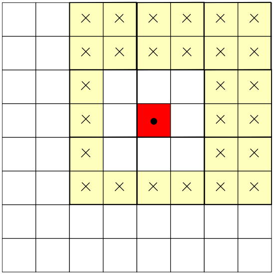
## The Mesh

### The Particle Mesh



# The Fast Multipole Algorithm
## The Mesh

Well separated cells



# The Fast Multipole Algorithm

## Expansions

- Multipole Expansions: A series, centered on a particular cell, which can approximate the potential from the cell's particles near that cell

# The Fast Multipole Algorithm

## Expansions

- Multipole Expansions: A series, centered on a particular cell, which can approximate the potential from the cell's particles near that cell
- Local Expansions: A series, centered on a particular cell, formed from a Multipole Expansion, which approximates the potential from within the cell at some distance away from that cell.

# The Fast Multipole Algorithm

## Aglorithm

- Form Multipole Expansions at the lowest mesh level

# The Fast Multipole Algorithm

## Aglorithm

- Form Multipole Expansions at the lowest mesh level
- Translate them to their parent's cells, combining the four children by summing them to form the parent's Multipole Expansion

# The Fast Multipole Algorithm

## Aglorithm



- Form Multipole Expansions at the lowest mesh level
- Translate them to their parent's cells, combining the four children by summing them to form the parent's Multipole Expansion
- Repeat the above for each level

# The Fast Multipole Algorithm

## Aglorithm

- Form Multipole Expansions at the lowest mesh level
- Translate them to their parent's cells, combining the four children by summing them to form the parent's Multipole Expansion
- Repeat the above for each level
- At the highest level, from the Multipole Expansion, form a Local Expansion.

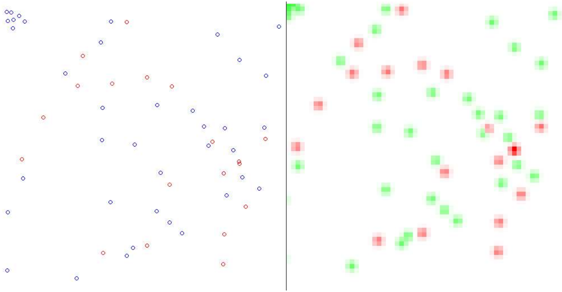
# The Fast Multipole Algorithm
## Aglorithm



- Form Multipole Expansions at the lowest mesh level
- Translate them to their parent's cells, combining the four children by summing them to form the parent's Multipole Expansion
- Repeat the above for each level
- At the highest level, from the Multipole Expansion, form a Local Expansion.
- Translate this to children nodes, repeat to the lowest mesh level.

## Basic structure of the Particle Mesh Ewald Method

- Interpolate particles to a grid



# Particle Mesh Ewald Method

## Basic structure of the Particle Mesh Ewald Method

Basic structure of the Particle Mesh Ewald Method

- Interpolate particles to a grid
- Perform a fourier transformation on the grid

Basic structure of the Particle Mesh Ewald Method

- Interpolate particles to a grid
- Perform a fourier transformation on the grid
- Calculate long range potential in reciprocal space using this grid

# Particle Mesh Ewald Method

## Basic structure of the Particle Mesh Ewald Method

Basic structure of the Particle Mesh Ewald Method

- Interpolate particles to a grid

- Perform a fourier transformation on the grid
- Calculate long range potential in reciprocal space using this grid
- Return to real space using another fourier transformation

Basic structure of the Particle Mesh Ewald Method

- Interpolate particles to a grid
- Perform a fourier transformation on the grid
- Calculate long range potential in reciprocal space using this grid
- Return to real space using another fourier transformation
- Calculate in real space directly for near particles, and use interpolated grid values for the long range potential.

# Particle Mesh Ewald Method
## Characteristics of the Particle Mesh Ewald Method

### Characteristics of the Particle Mesh Ewald Method

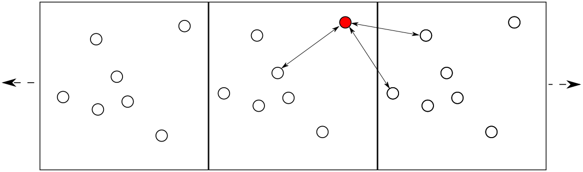
- Complexity of $O(n\log(n))$

# Particle Mesh Ewald Method

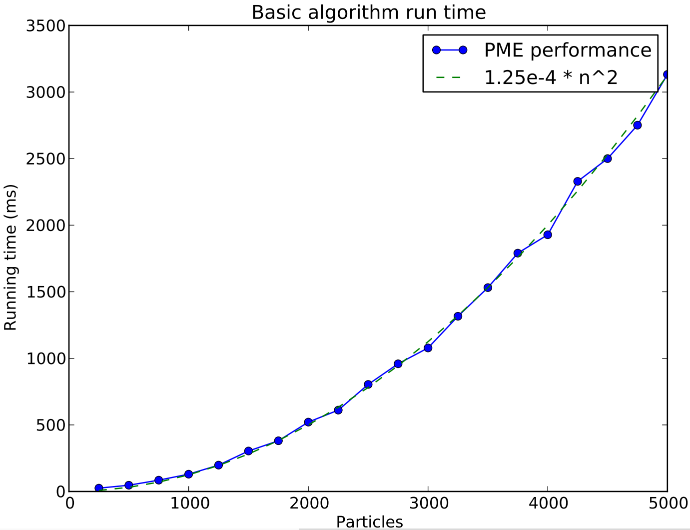
## Characteristics of the Particle Mesh Ewald Method

Characteristics of the Particle Mesh Ewald Method

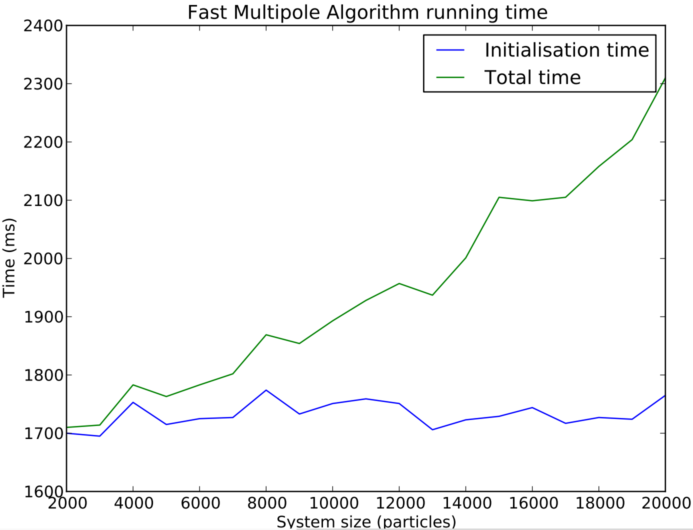
- Complexity of $O(n\log(n))$
- Fourier transformation technique leads to Periodic Boundary Conditions



# Results
## Basic Algorithm

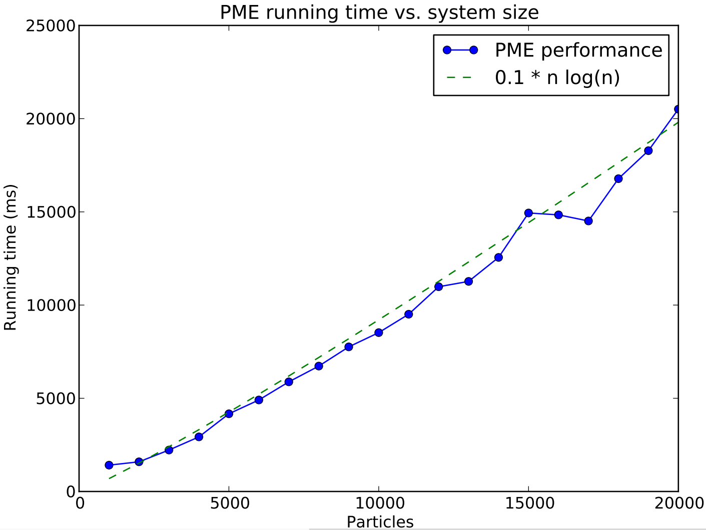


# Results
## Fast Multipole Algorithm



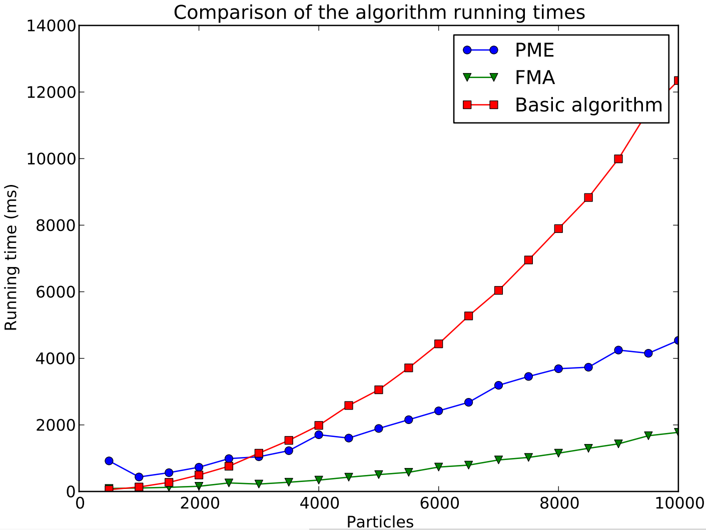
# Results
## Particle Mesh Ewald Algorithm





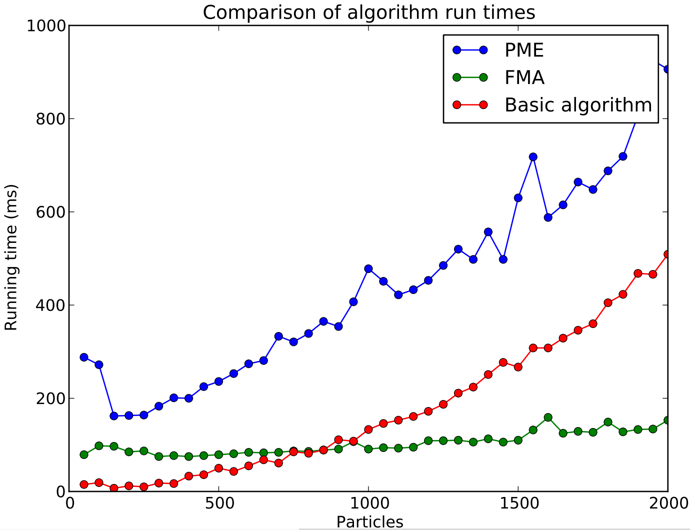
# Results
## Comparison





# Results
## Comparison





# Conclusion
## Methods



Joshua Nelson, u4850020



- Fast Multipole Algorithm is the fastest with the lowest run times over most cases

# Conclusion
## Methods



- Fast Multipole Algorithm is the fastest with the lowest run times over most cases
- Particle Mesh Ewald has desriable features and more potential for paralellisation and optimisation

# Conclusion
## Methods



- Fast Multipole Algorithm is the fastest with the lowest run times over most cases
- Particle Mesh Ewald has desriable features and more potential for paralellisation and optimisation
- The Basic solution is not feasible for very large system sizes

# Conclusion

## Contributions





- Created fucntional implementations of the Basic, Particle Mesh Ewald, and Fast Multipole algorithms in Java.

# Conclusion

## Contributions





- Created fucntional implementations of the Basic, Particle Mesh Ewald, and Fast Multipole algorithms in Java.
- Well documented and coded with readability in mind

# Conclusion
## Contributions



- Created fucntional implementations of the Basic, Particle Mesh Ewald, and Fast Multipole algorithms in Java.
- Well documented and coded with readability in mind
- Actualization of often abstractly defined algorithms

# Conclusion

## Contributions

- Created fucntional implementations of the Basic, Particle Mesh Ewald, and Fast Multipole algorithms in Java.
- Well documented and coded with readability in mind
- Actualization of often abstractly defined algorithms
- Complexity analysis of algorithms, crossover points found.

# Conclusion

## Contributions



- Created fucntional implementations of the Basic, Particle Mesh Ewald, and Fast Multipole algorithms in Java.
- Well documented and coded with readability in mind
- Actualization of often abstractly defined algorithms
- Complexity analysis of algorithms, crossover points found.
- Bridged the gap between mathematicians, computer scientists, and computational chemists.