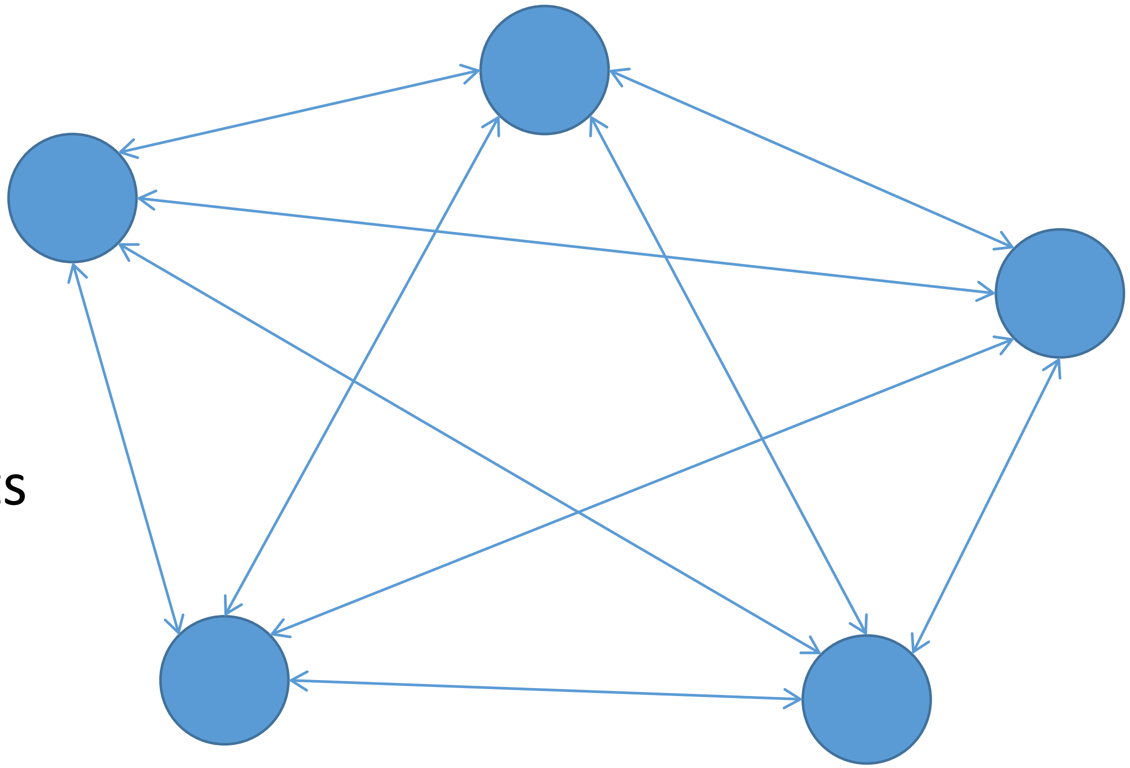
# Solarium

A solar system simulator

Peter Chapin

# N-Body Problem



N interacting objects
*N(N-1)/2* edges
*O($N^2$)* interactions

# Solarium

- Simulates movement of solar system objects
  - Considers mutual gravitational interaction
  - Supports many (thousands) of objects
    - Planets
    - Asteroids
    - Comets
    - Bowling balls
- Each object represented by a "dynamics"
  - Position (x, y, z components)
  - Velocity (x, y, z components)

# Time

- Time is divided into “time steps”
- For each time step…
  - For each object…
    - New position based on old position and old velocity
    - New velocity based on old velocity and current acceleration
    - Acceleration is calculated by…
      - Computing total force on an object due to gravitational attraction of all other objects ($O(N^2)$)
      - $F = ma$ so $a = F/m$
- Repeat for multiple time steps
  - If a time step is one hour, there are 8766 time steps per simulated year

# Inaccurate

- In real life time is continuous
  - Velocity changes smoothly over a time step
- In our simulation velocity is constant during a time step
  - … makes an abrupt change at the end of the time step
- We can increase accuracy by making shorter time steps
  - Particularly important when velocity changes rapidly
  - … such as during a close approaches between two objects
- **Small time steps greatly increase computation time**
  - *Trade off between computation time and accuracy!*

# Ideally…

- We should understand what accuracy we need…
  - … and then compute no harder than necessary

# More Issues

- What we are doing is essentially numerical integration
  - Many algorithms for this are known
- Advanced integration methods use past history to estimate the future
  - … can give reasonable accuracy with courser time steps
  - … but require more computation to make the estimates
- *Ideally we would use a numerical integration method that optimizes computation time while respecting desired accuracy*
  - This is a subject for a future version of the program

# Still More Issues

- Our algorithm, "*All Pairs*," is $O(N^2)$
- Better algorithms exist for solving the N-body problem!
    - Barnes Hut runs in $O(N \log(N))$ time
        - A huge improvement!
    - … but is not 100% accurate
    - This adds another wrinkle to the accuracy vs computation time trade off
- A serious computation would…
    - Use the best algorithms available
    - Compute an answer <u>as inaccurately</u> (i. e., as quickly) as possible
- *We are interested in parallel programming so these points are minor*