N-Body Modelling

Computational Astrophysics 2016

F. H. Panther

Overview

- Introduction to some techniques in N-body modelling of collisional systems and direct N-body codes
- Brief summary of GPU acceleration using CUDA/C ++
- NOT: advanced algorithms, coding
- Based on lectures by Michela Mapelli

What is an N-body simulation?

- numerical integration of forces acting on N bodies for a time t
- e.g. astrophysics, fluid dynamics, molecular dynamics
    - In astrophysics? F = GRAVITY

Simple physics!

$$\ddot{\vec{r}}_i = -G \sum_{j \neq i} m_j \frac{\vec{r}_i - \vec{r}_j}{|\vec{r}_i - \vec{r}_j|^3}$$

1687: Newton derives eqns

Written equivalently as 6N coupled differential equations

$$\begin{cases} \dot{v}_i = -\sum \dfrac{G\, m_j}{r_{ij}^3} x_{ij} \\ \dot{x}_i = v_i \end{cases}$$

Can the problem be solved analytically?

1710: Bernoulli derives solution for N = 2

Simple physics?

- Analytic solutions for N = 2 and restricted cases of N=3
- Problem demands a numerical solution - What is the best way to numerically integrate equations?

$$\begin{cases} \dot{v}_i = -\sum \dfrac{G\, m_j}{r_{ij}^3} x_{ij} \\ x_i = v_i \end{cases}$$

Solving the N-body problem

Write down the Taylor expansion:

$$x_i(t + \Delta t) = x_i(t) + \frac{dx_i(t)}{dt}\Delta t + \frac{1}{2}\frac{d^2x_i(t)}{dt^2}\Delta t^2 + O(\Delta t^3)$$

$$v_i(t + \Delta t) = v_i(t) + \frac{dv_i(t)}{dt}\Delta t + \frac{1}{2}\frac{d^2v_i(t)}{dt^2}\Delta t^2 + O(\Delta t^3)$$

Predict velocity and position at t + dt using current information

Truncation error: information about "order" of method

Common numerical integration techniques are based on the Taylor expansion

Euler integrator

- The simplest numerical integrator
- “Explicit” - depends only on information we know at time t
- 2nd order method:

$$x_i(t + \Delta t) = x_i(t) + \frac{dx_i(t)}{dt}\Delta t + O(\Delta t^2)$$

$$v_i(t + \Delta t) = v_i(t) + \frac{dv_i(t)}{dt}\Delta t + O(\Delta t^2)$$

Euler integrator

- While the Euler integrator is simple, it suffers from major limitations - what are they?
- Energy conservation!

- Fun activity: Write a simple 2-body code to simulate two stars of equal mass orbiting each other, calculate the KE of the system. Is it conserved?

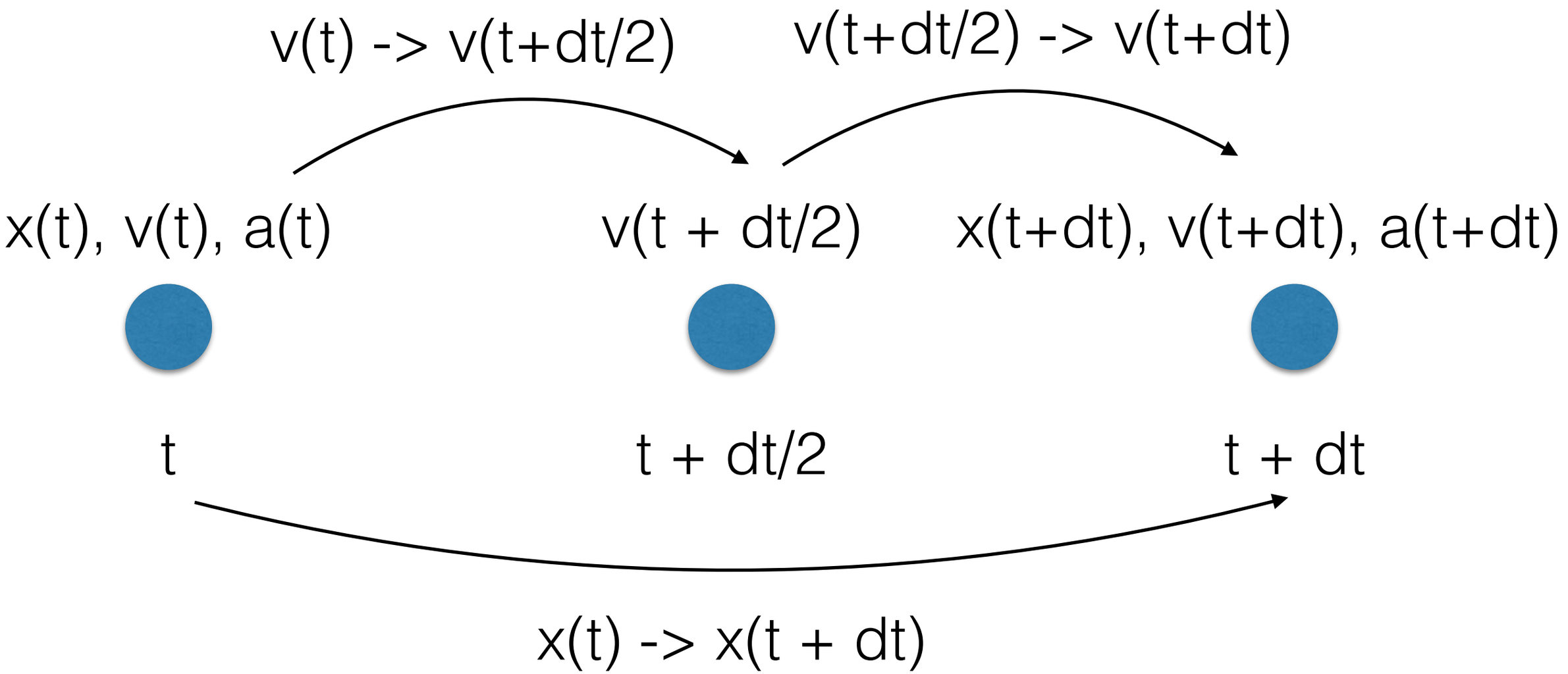
Leapfrog integrator

- Also second order, but conserves energy far better than Euler

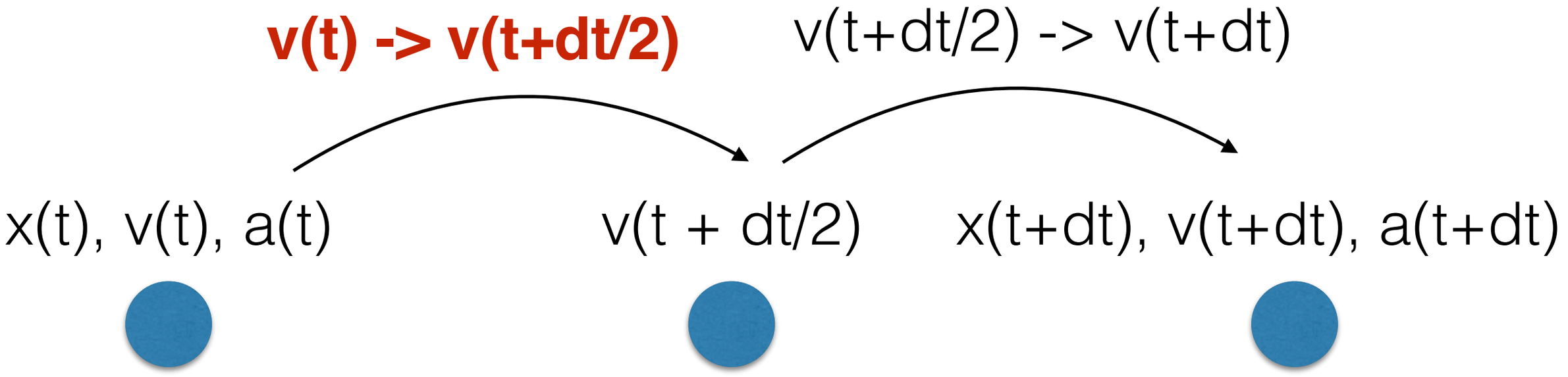
- Position and velocity evaluation “leapfrog” each other within timesteps

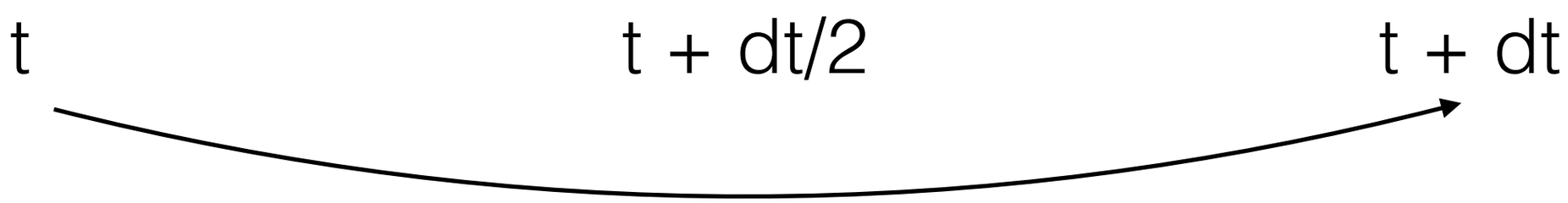
- Symplectic (has global stability)

Leapfrog integrator



Leapfrog integrator

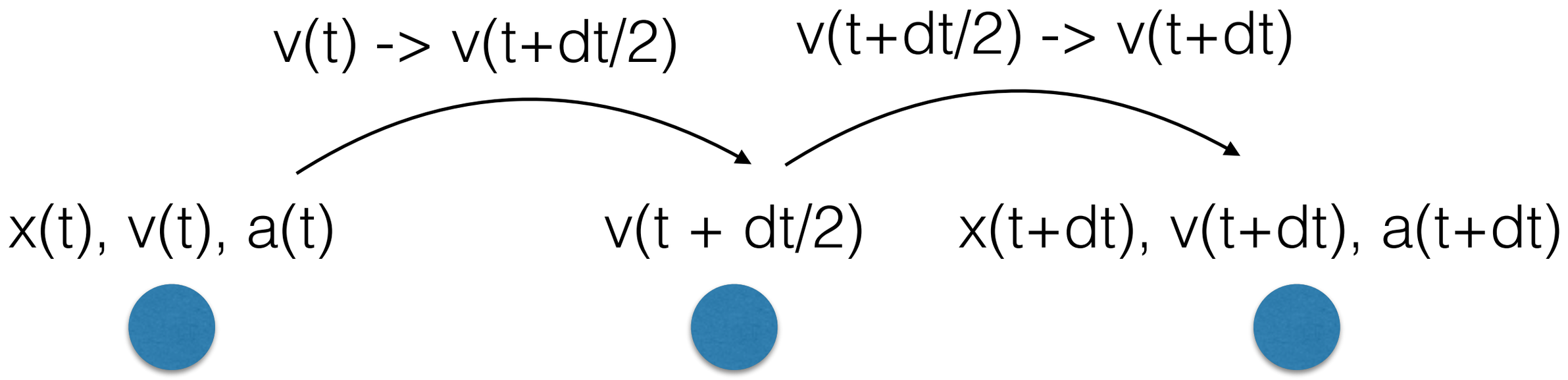


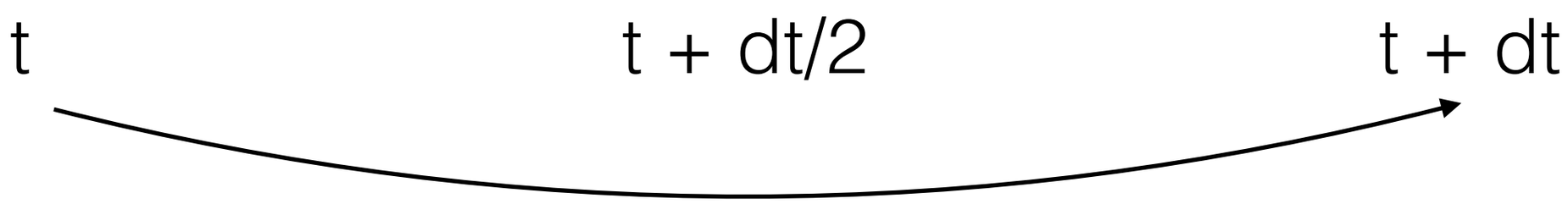


x(t) -> x(t + dt)

$$v(t + \frac{dt}{2}) = v(t) + \frac{dt}{2}a(t)$$

Leapfrog integrator

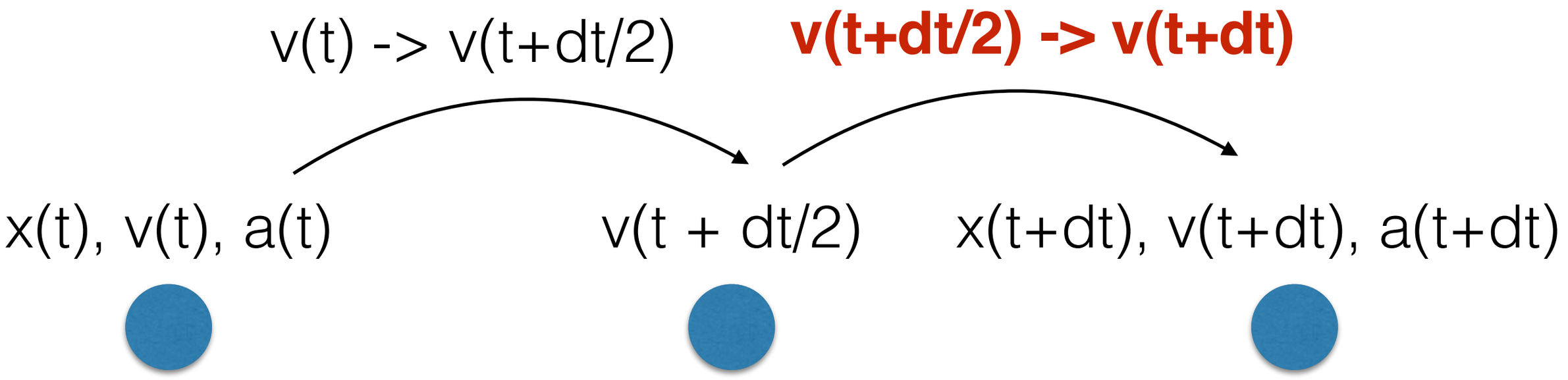


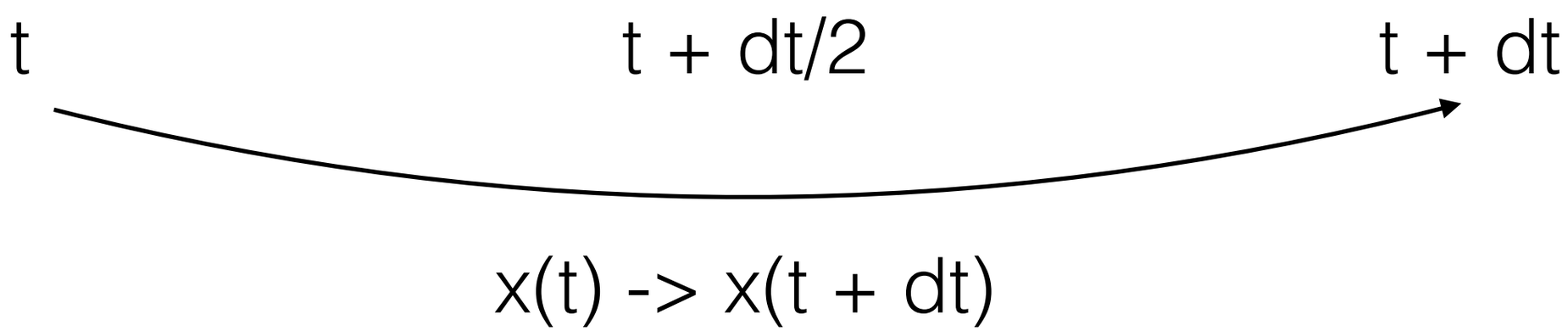


x(t) -> x(t + dt)

$$x(t + \frac{dt}{2}) = x(t) + v(t + \frac{dt}{2})dt$$

Leapfrog integrator





$$v(t+dt) = v(t+\tfrac{dt}{2}) + \tfrac{1}{2}a(t+dt)dt$$

Leapfrog integrator

- Putting it all together:

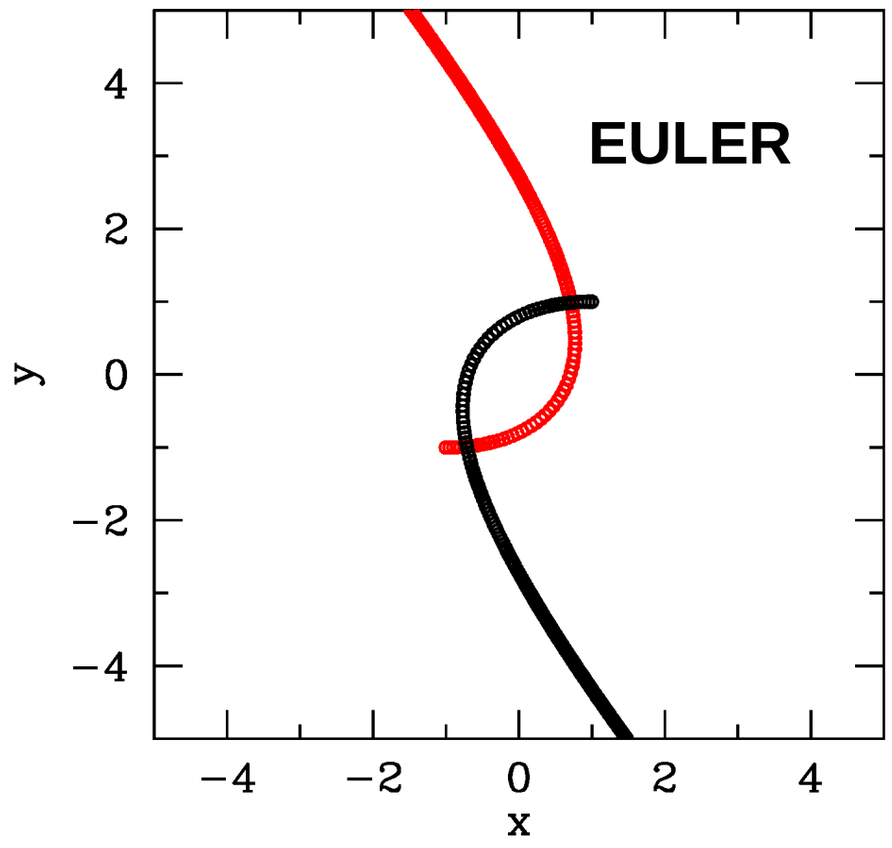
$$v(t+dt) = v(t) + \frac{1}{2}a(t)dt + \frac{1}{2}a(t+dt)dt$$

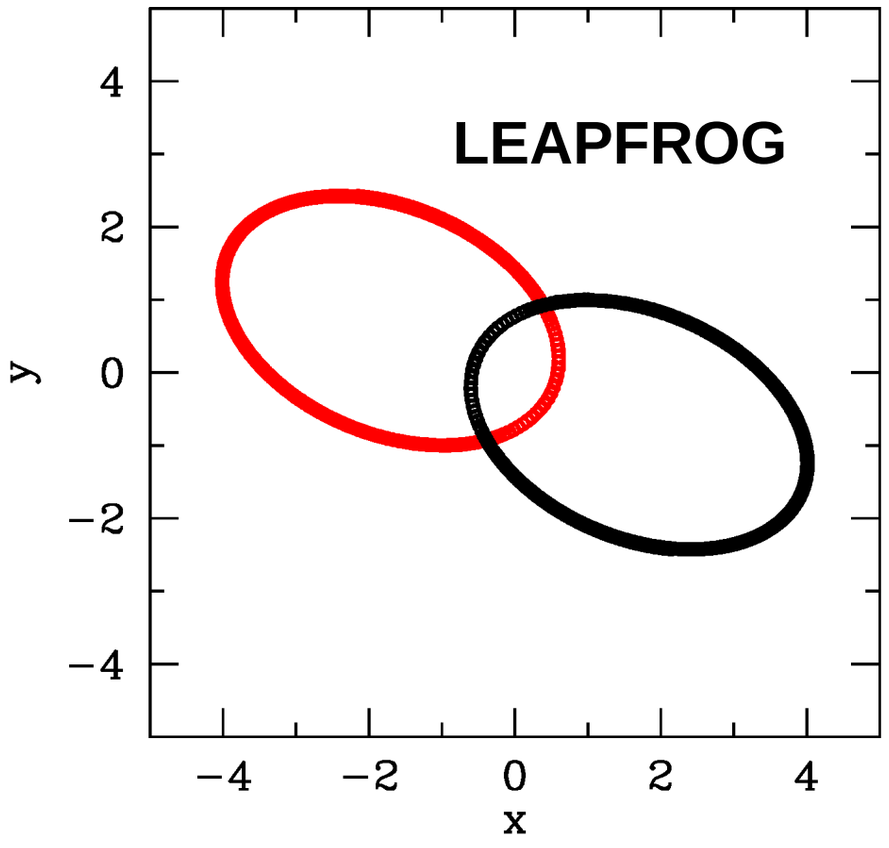
$$x(t+dt) = x(t) + v(t)dt + \frac{1}{2}a(t)dt^2$$

- Fun activity part two: integrate the same 2 body problem as you did with the Euler integrator and see what happens

2-body system

Spoiler alert





Complexity

- How many calculations do you perform for N particles?
- Complexity grows as N^2 - quickly
- Reducing complexity is not always the best choice: I am going to talk about collisional systems, in which we cannot reduce complexity
- In collisionless simulations, there are several different ways to reduce complexity - look up “Barnes-Hut tree method” and “multipole expansion”. These methods are used in cosmological simulations

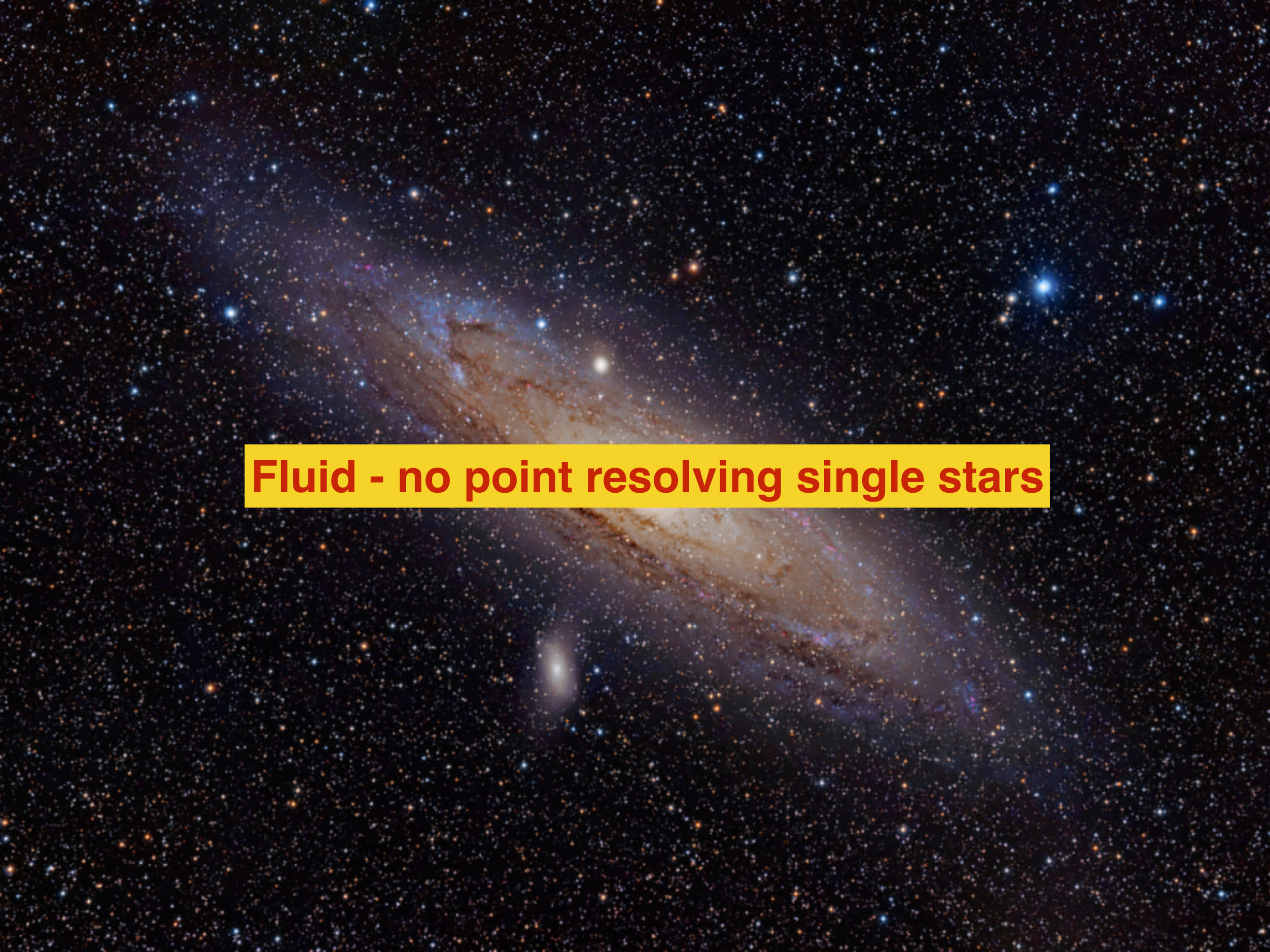
Direct N-body codes

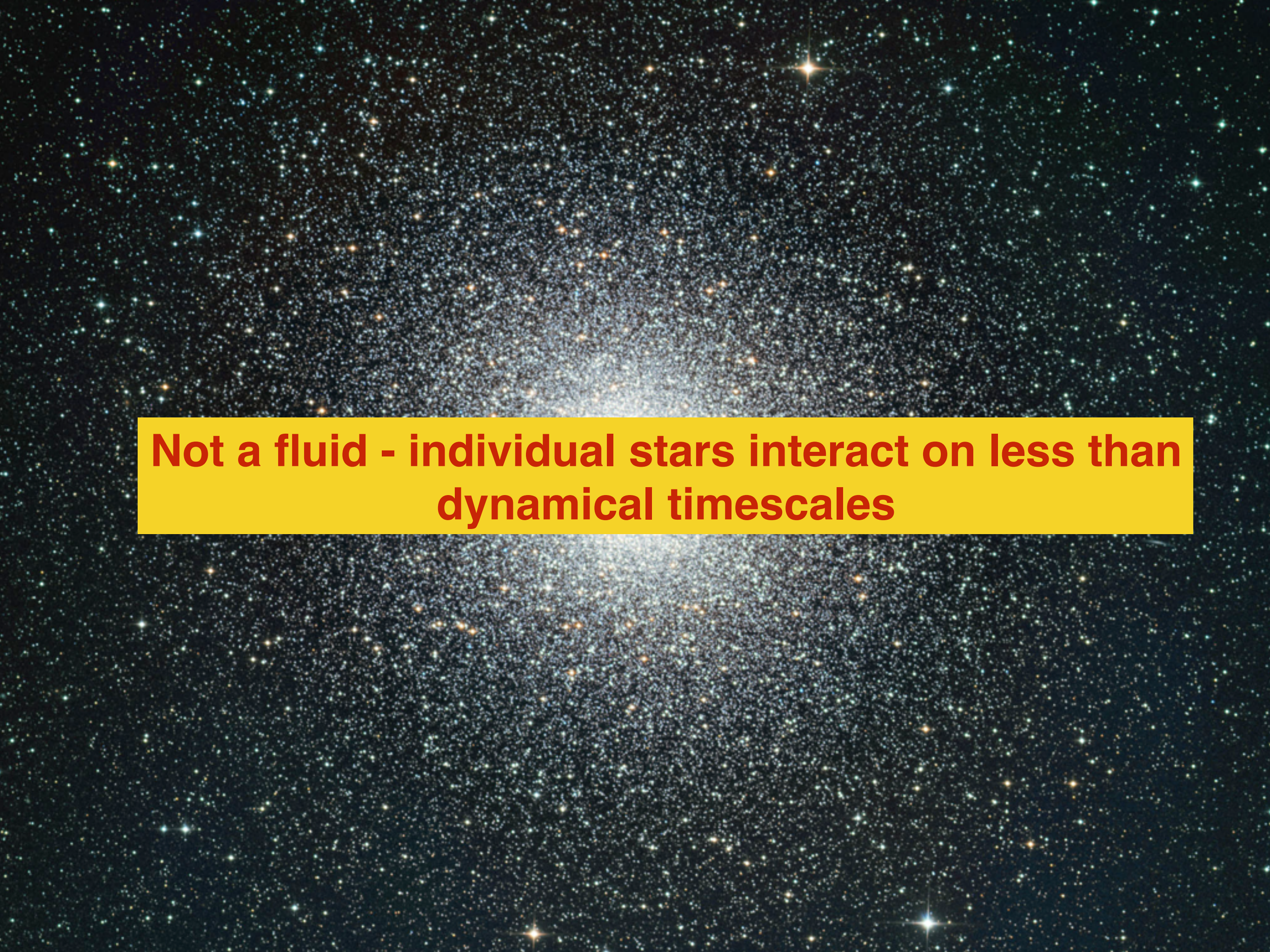
Only consider gravity - no sub-grid physics

$$\ddot{\vec{r}}_i = -G \sum_{j \neq i} m_j \frac{\vec{r}_i - \vec{r}_j}{|\vec{r}_i - \vec{r}_j|^3}$$

Direct N-body codes calculate all of the N^2 forces between particles, rather than ignoring forces between most distant particles

To do so many calculations is expensive, so why?


Fluid - no point resolving single stars


Not a fluid - individual stars interact on less than dynamical timescales

Stars mind their own business (except in the NSC)

Many stars are in binaries, but varying the binary fraction has no impact on dynamics (you don't resolve dynamics on single-star level)

boring

Stars mostly mind their own business, but interactions are still common

Initial binary fraction can have a significant influence on cluster dynamics

interesting

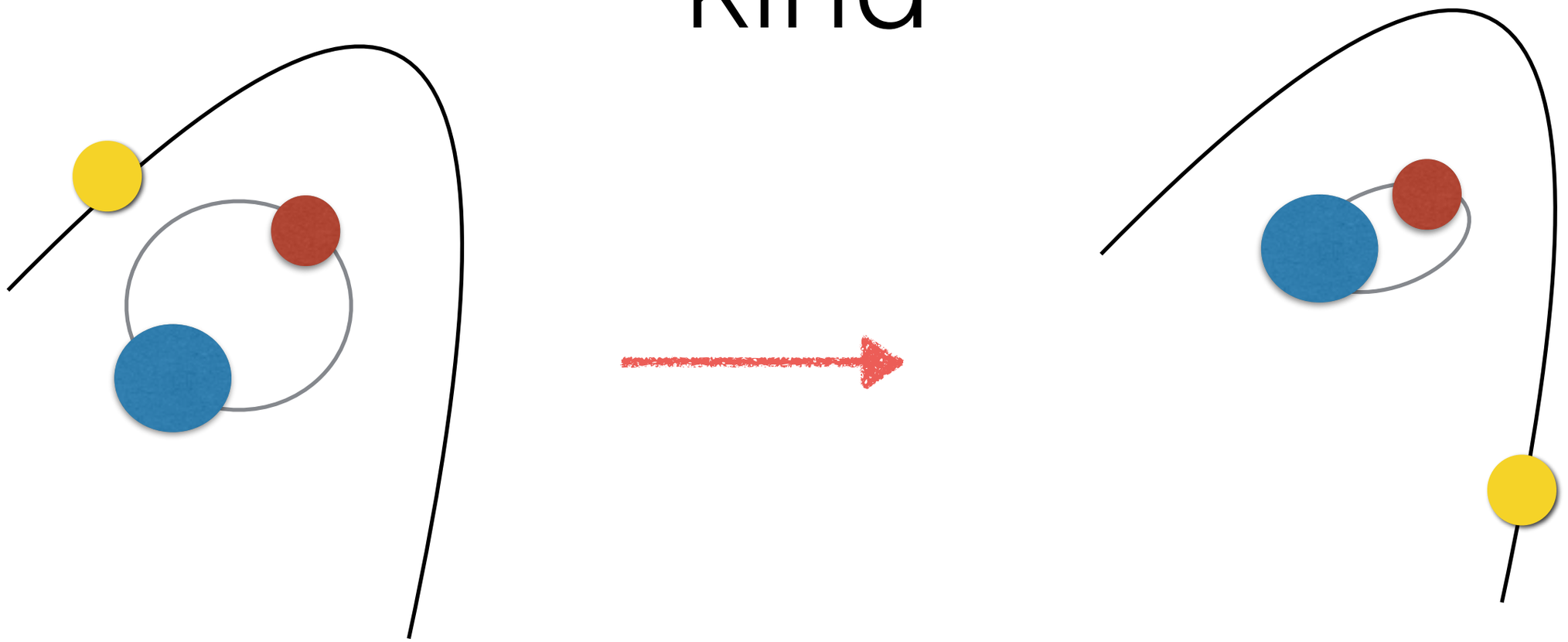
Direct N-body codes

- Direct N-body codes are used to investigate regions of concentrated high stellar density: Globular clusters, nuclear star clusters and young clusters

- In these regions, 3 body encounters are common

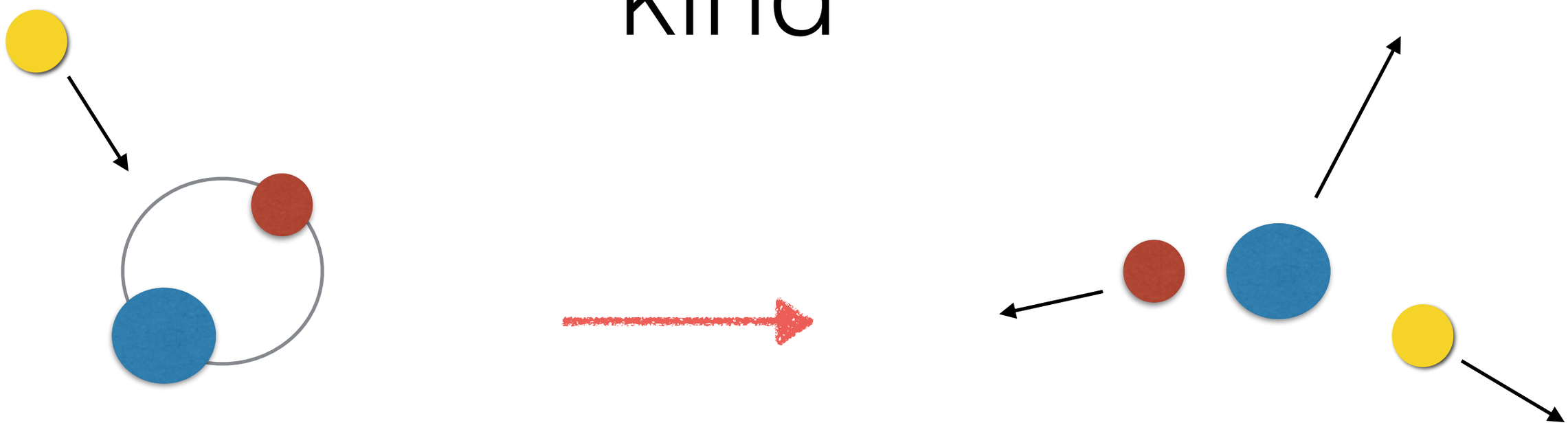
- Binary systems interact with single stars - **energy is exchanged**

Close encounters of the 3-body kind



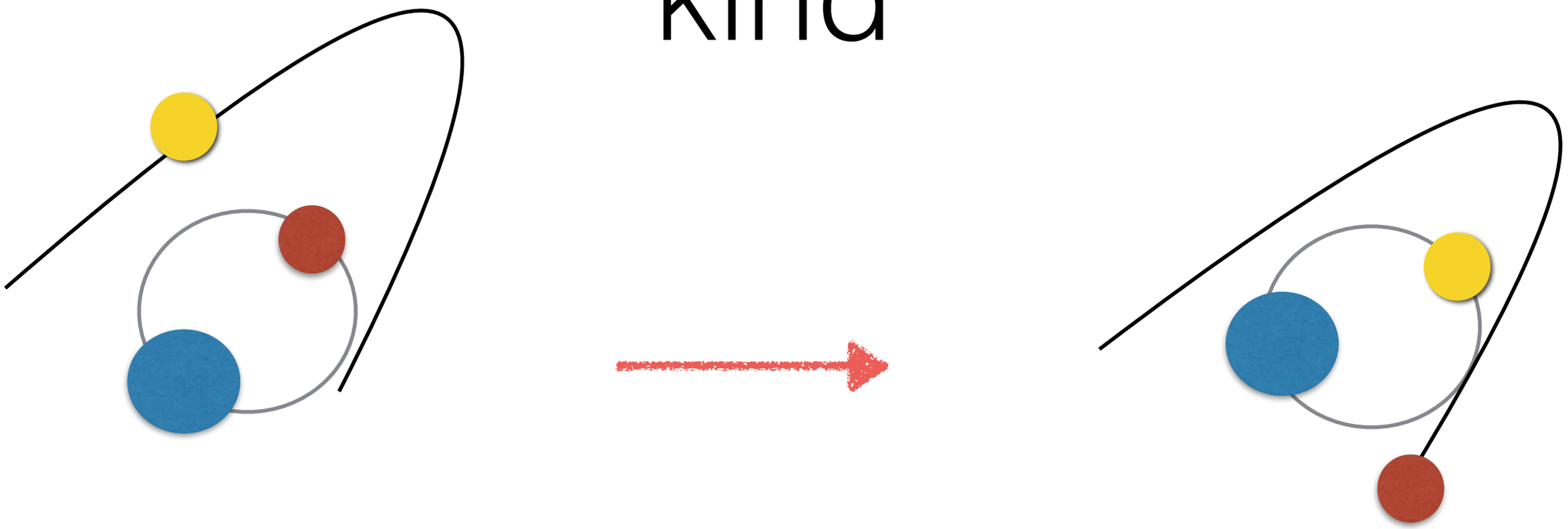
Fly-by encounters change the orbits of the central binary system

Close encounters of the 3-body kind



"Ionization" destroys the central binary

Close encounters of the 3-body kind



Particle exchange sees one star kicked out of the binary and replaced with another

Close encounters of the 3-body kind

- Integrating these encounters correctly is a major computational challenge, and accounts for most of the difficulty in developing direct N-body codes

- Require small integration steps (~ a few years physical time) and high order integration schemes with low errors to conserve energy and angular momentum

- Need to be able to:

    - integrate perturbations on <dynam… scale (less than 1 orbit)

    - Conser… and angular momentum well

Use 4th order Hermite Gauss integration

Hermite Gauss Integration

- High-order Hermite-Gauss integration is ubiquitous in direct N-body codes

- Most use a 4th order scheme (NBODY6 (Aarseth +2006) uses 6th order)

- Use the jerk (time derivative of acceleration) and a predictor-corrector scheme

$$j = \frac{da}{dt} = G \sum M_j [\frac{v_{ji}}{r_{ji}^3} - 3 \frac{(r_{ji} v_{ji}) r_{ji}}{r_{ji}^5}]$$

Hermite Gauss Integration

But wait, there's more - add a "softening" term (physical radius of stars)

$$a_i = G\sum \frac{M_j r_{ij}}{(r_{ji}^2+\epsilon^2)^{5/2}}$$

$$j = G\sum M_j\left[\frac{v_{ij}}{(r_{ji}^2+\epsilon^2)^{3/2}} + \frac{3(v_{ij}r_{ij})r_{ji}}{(r_{ji}^2+\epsilon^2)^{5/2}}\right]$$

Now we're ready to look at the integration scheme

Hermite Gauss Integration

Start off with the Taylor expansion for all the terms we want to know

$$
\begin{cases}
x_1 = x_0 + v_0\,\Delta t + \frac{1}{2}\,a_0\,\Delta t^2 + \frac{1}{6}\dot{j}_0\,\Delta t^3 + \frac{1}{24}\ddot{j}_0\,\Delta t^4 & (1)\\
v_1 = v_0 + a_0\,\Delta t + \frac{1}{2}\dot{j}_0\,\Delta t^2 + \frac{1}{6}\ddot{j}_0\,\Delta t^3 + \frac{1}{24}\dddot{j}_0\,\Delta t^4 & (2)\\
a_1 = a_0 + j_0\,\Delta t + \frac{1}{2}\ddot{j}_0\,\Delta t^2 + \frac{1}{6}\dddot{j}_0\,\Delta t^3 & (3)\\
j_1 = j_0 + \dot{j}_0\,\Delta t + \frac{1}{2}\ddot{j}_0\,\Delta t^2 & (4)
\end{cases}
$$

4th order accuracy, but only contains 2nd order terms!

$$x_1 = x_0 + \frac{1}{2}\,(v_0 + v_1)\,\Delta t + \frac{1}{12}\,(a_0 - a_1)\,\Delta t^2 + O(\Delta t^5)$$

$$v_1 = v_0 + \frac{1}{2}\,(a_0 + a_1)\,\Delta t + \frac{1}{12}\,(j_0 - j_1)\,\Delta t^2 + O(\Delta t^5)$$

Hermite Gauss Integration

Predictor - corrector: These equations contain implicit terms, so we need something to predict them. This is the Predictor step

We use the 3rd order Taylor expansion to do so

$$x_{p,1} = x_0 + v_0\,\Delta t + \frac{1}{2}\,a_0\,\Delta t^2 + \frac{1}{6}\,j_0\,\Delta t^3$$

$$v_{p,1} = v_0 + a_0\,\Delta t + \frac{1}{2}\,j_0\,\Delta t^2$$

These predictions are used to calculate the predicted acceleration and jerk from Newton's formula. This step is called the Force Evaluation

Hermite Gauss Integration

The predicted values from the force evaluation are substituted into the starting equations

$$x_1 = x_0 + \frac{1}{2}\left(v_0 + v_{p,1}\right)\Delta t + \frac{1}{12}\left(a_0 - a_{p,1}\right)\Delta t^2$$

1 $$v_1 = v_0 + \frac{1}{2}\left(a_0 + a_{p,1}\right)\Delta t + \frac{1}{12}\left(j_0 - j_{p,1}\right)\Delta t^2$$

What order is this in position?

3rd - how do we make this 4th order?

Evaluate this first, and then substitute it into the position equation (a kind of dirty trick)

Timesteps

- What sort of timestep would you choose?
    - Same for all particles?
    - Long?
    - Very short?

$$\Delta t_i = \eta \frac{a_i}{\dot{j}_i}$$

Timesteps

- Calculate the timestep for some particle i using $\Delta t_i = \eta \frac{a_i}{\dot{j}_i}$
- Simulation time is set to $t = t_i + \min(dt_i)$
- All particles with the this timestep are called active particles - the predictor-corrector steps are done for these active particles
  - Positions and velocities predicted for all particles, but acceleration and jerk are calculated only for the active particles
  - positions and velocities are corrected for the active particles
  - Everything starts over and repeats

Unfortunately, this is expensive and the system will lose coherence - block time steps can be a good choice

Regularization

- What is it?
  - A way to remove the singularity in Newton's equations when stars come infinitely close together
  - But didn't we deal with that with the softening term?
    - No - softening adds the physical radius of stars but does not remove the singularity. Regularisation employs a change in coordinate systems.
  - Common regularizations are KS (Kustaanheimo-Steifel) regularization (for 3-body and binary encounters) and Aarseth/Mikkola CHAIN regularization

Acceleration

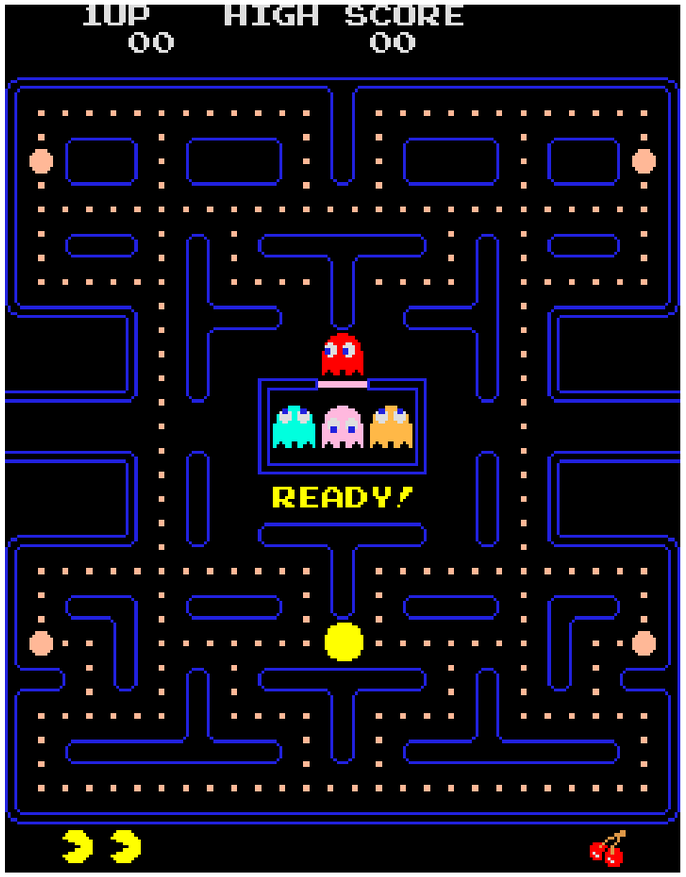
- N-body calculations require a large number of floating point operations - the bigger N is, the bigger this number

- While not “trivially” parallel, direct N-body is naturally parallelizable:

    - Single instruction, multiple data (SIMD)

    - Generating computer graphics for modern video games works on a similar principle!

Acceleration



Creating modern video-game environments involves mapping pixel colors and transparencies to individual points - this is a SIMD process

Acceleration

- The first N-body accelerator used GRAPE (GRAvity PipE) - highly specialized hardware (1989-2001)

- Treats pairwise force calculations and accelerates this process much in the way a graphics card accelerates graphics processing.

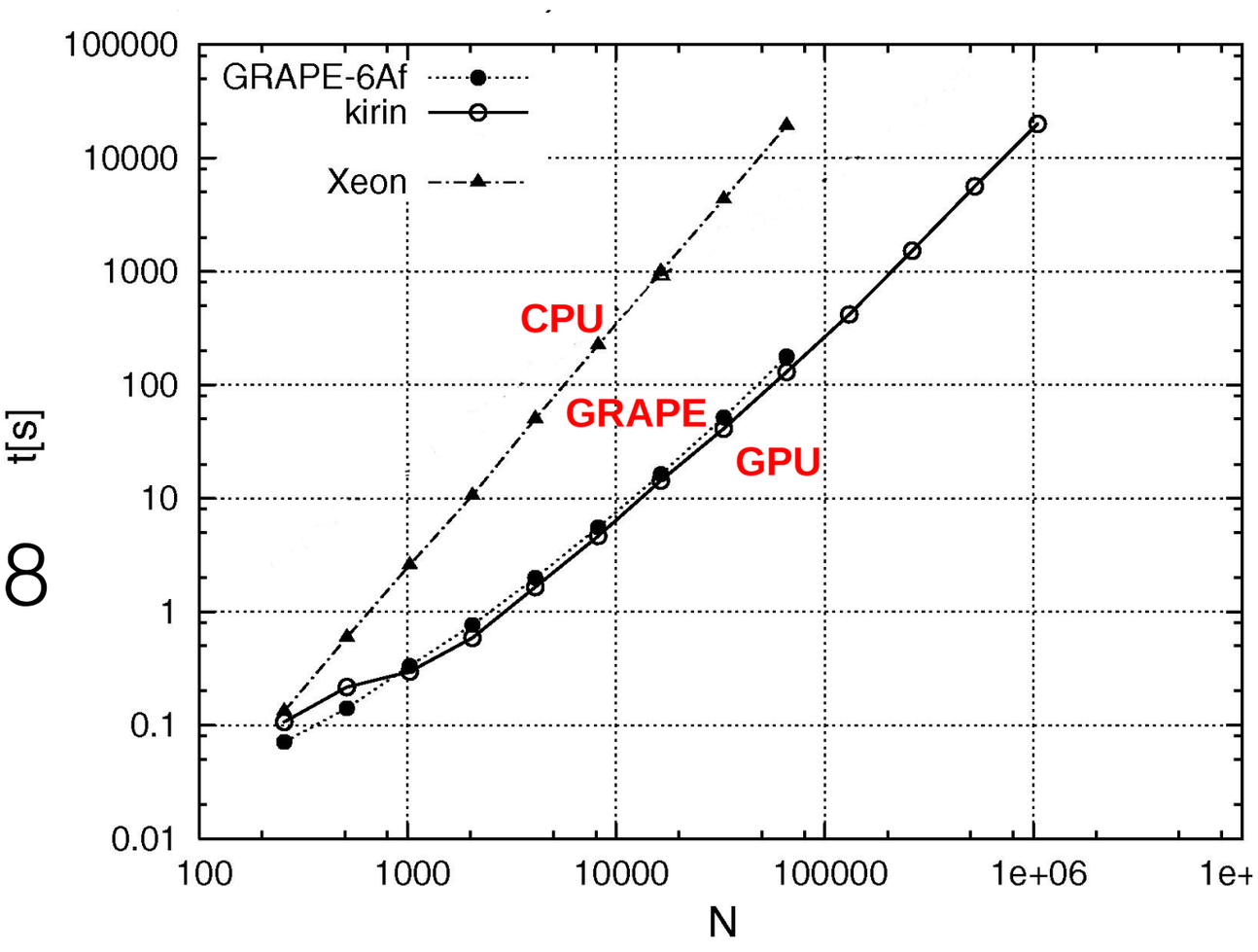
- Hetrogenous: Predictor-corrector step on PC, acceleration and jerk (gravity step) on GRAPE

Acceleration

- The GRAPE-4 was a 4-cabinet computer that was the first to reach 1Tflop

- In 2004, GRAPE was superseded by modern GPUs, which were just as fast as GRAPE for direct N-body

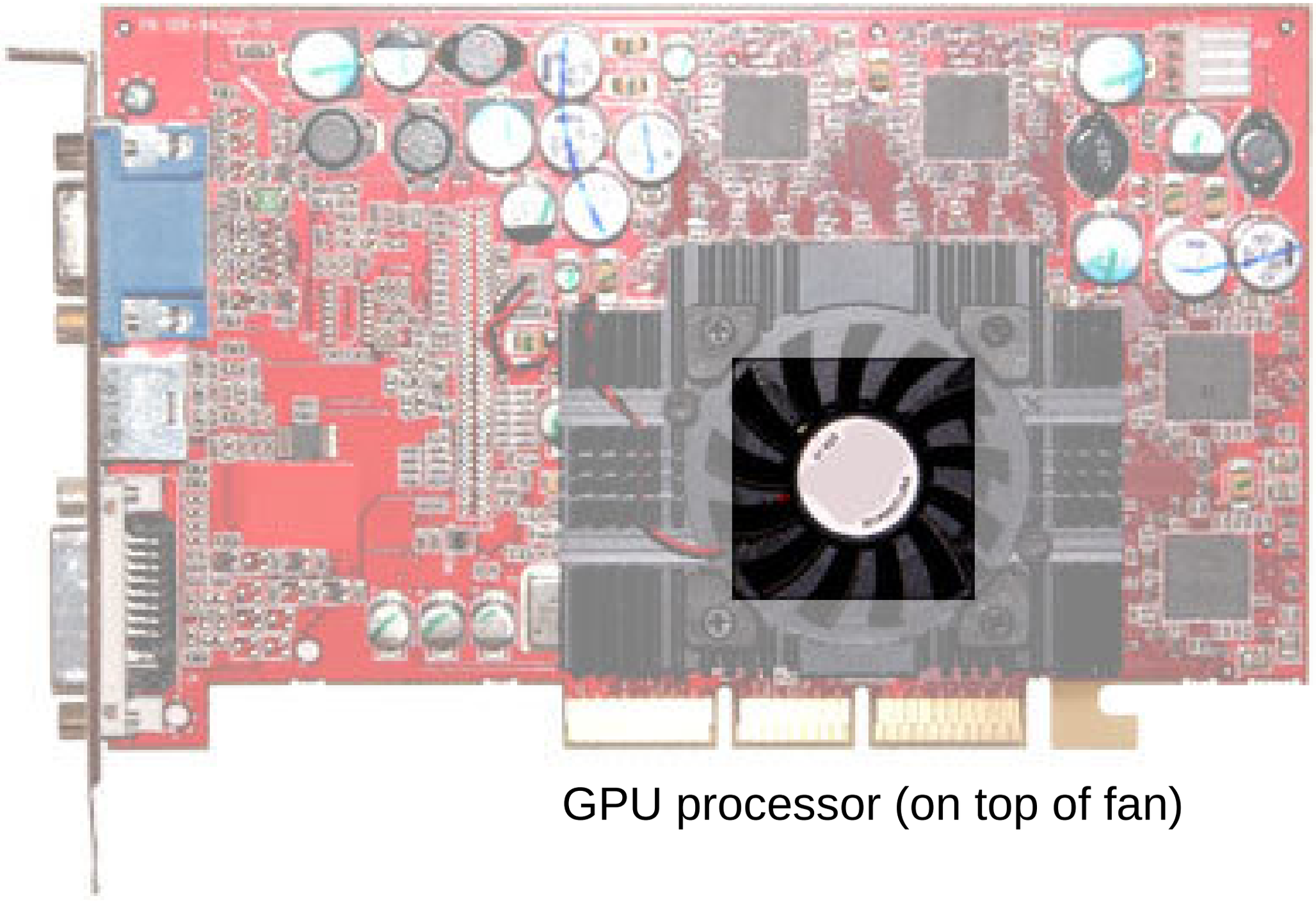
Belleman+2008



Acceleration

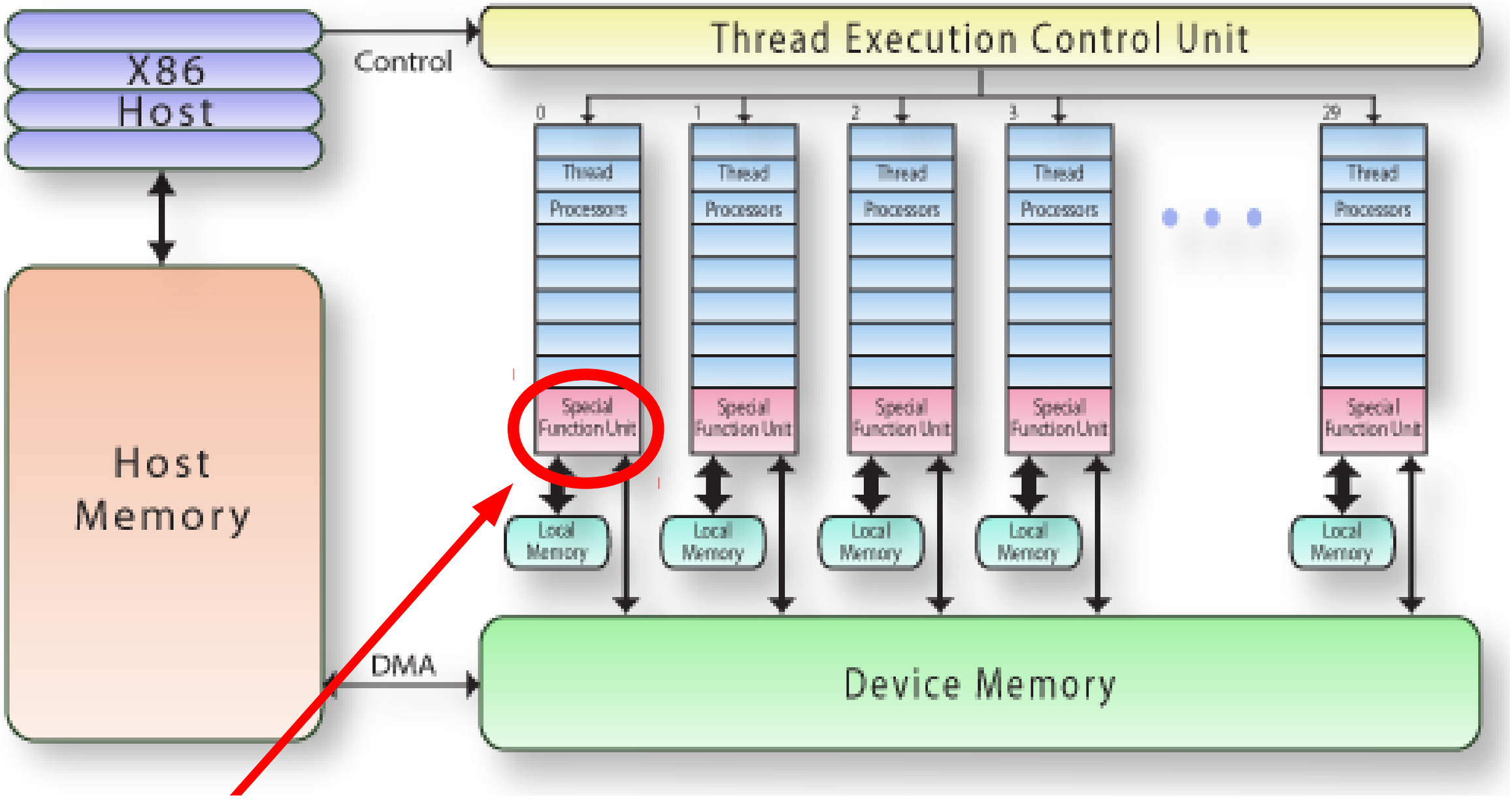
- First N-body simulations on GPU by Nyland+2004
- First GPU implementation of Hermite integration (Portegies-Zwart+2007)
- Now preferred for direct N-body calculations


NVIDIA TESLA



GPU processor (on top of fan)

see http://www.tomshardware.com/reviews/graphics-beginners,1288-2.html for an excellent overview!


X86
Host
Control
Thread Execution Control Unit
0
1
2
3
29
Thread
Processors
Special
Function Unit
Local
Memory
Host
Memory
DMA
Device Memory