GPUnit

Daniel Bagnell
Jason Economou
Rajkumar Jayachandran
Tim McJilton
Gabriel Schwartz
Andrew Sherman

Advisor: Prof. Jeremy Johnson
Stakeholders: Prof. Steve McMillan
Alfred Whitehead
The Leiden Observatory

May 2, 2011

Overview

Motivation

- Astrophysics researchers need to simulate star clusters and galaxies.
- Every star pulls on all of the others: $O\left(n^2\right)$ for the naive case.
- Stars evolve over time, mass changes.

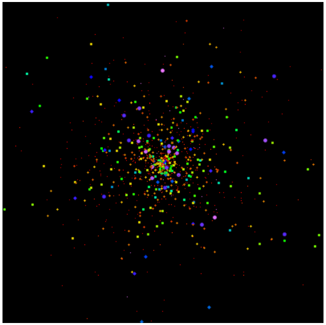


Figure: N-Body Simulation: 1024 Stars

http://www.sns.ias.edu/~starlab/animations/

Astrophysical Multipurpose Software Environment (AMUSE)

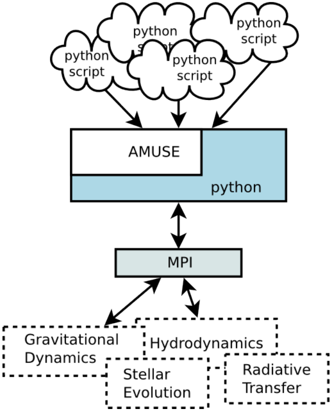


Figure: AMUSE Architecture

State of AMUSE

- Currently used by researchers to run large-scale simulations.
- Scripts, diagnostics, logging are all written by hand.
- AMUSE API/programming knowledge is required to create experiments.

Purpose of GPUnit

- Ease the use of AMUSE
- Create/Design/Modify experiments
- Select, configure, swap out modules and initial conditions
- Store and restore progress of running experiments.

Target Audiences

- Physics Students
- Observational Astrophysicists
- Theoretical Astrophysicists

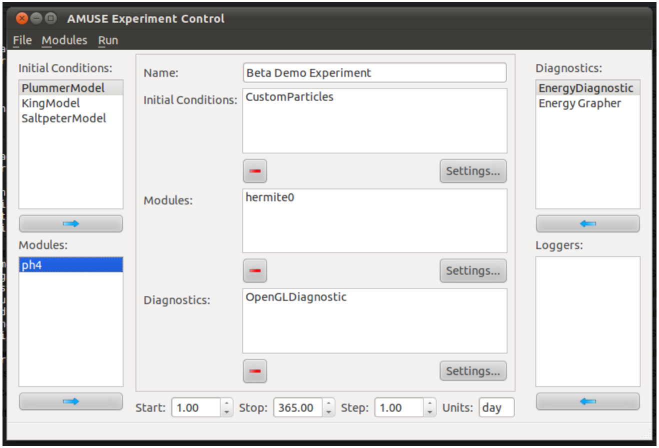
Features and Design

- Configurable experiments that can be saved and shared.
- Diagnostic tools that compute useful metrics.
- Storage of experiment state in case of crashes.
- Interface for custom diagnostics and code.

Features and Design

- Written in Python using the PyQt4 GUI toolkit.
- AMUSE is written in Python, improves interaction.
- Provides a display of cluster usage to aid in scheduling.

Experiment Editor


AMUSE Experiment Control

Initial Conditions:
PlummerModel
KingModel
SaltpeterModel

Modules:
ph4

Name: Beta Demo Experiment
Initial Conditions: CustomParticles
Modules: hermite0
Diagnostics: OpenGLDiagnostic

Start: 1.00 Stop: 365.00 Step: 1.00 Units: day

Diagnostics:
EnergyDiagnostic
Energy Grapher

Loggers:

Figure: Experiment Editor

Cluster Control

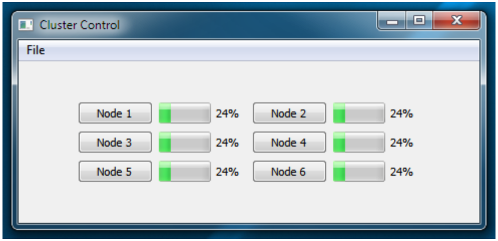
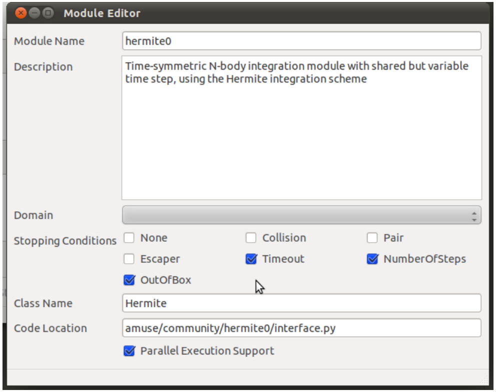


Figure: Cluster View

Module Specification



Module Name: hermite0

Description: Time-symmetric N-body integration module with shared but variable time step, using the Hermite integration scheme

Domain:

Stopping Conditions: ☐ None ☐ Collision ☐ Pair ☐ Escaper ☑ Timeout ☑ NumberOfSteps ☑ OutOfBox

Class Name: Hermite

Code Location: amuse/community/hermite0/interface.py

☑ Parallel Execution Support

Figure: Module Editor

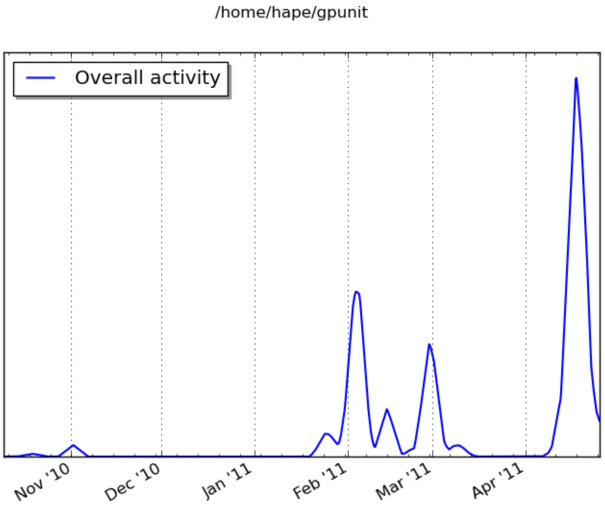
Tests

- Table of tests that pass.

Team Management

- Used Mercurial as our version control system.
  - Distributed, allows offline commits.
- Team met weekly.
  - Once to plan work, once to code.
- Bi-weekly advisor meetings.

Commit History



Demo

- Demonstrate a simulation from start to finish here.

Questions

- Questions?