

# High Performance Computing

---

What is it used for and why?



EPSRC



NERC SCIENCE OF THE ENVIRONMENT



archer



CRAY  
THE SUPERCOMPUTER COMPANY



epcc



# Overview

- What is it used for?
  - Drivers for HPC
  - Examples of usage
- Why do you need to learn the basics?
  - Hardware layout and structure matters
  - Serial computing is required for parallel computing
  - Appreciation of fundamentals will help you get more from HPC and scientific computing



# What is HPC used for?

Drivers and examples



# Why HPC?

- Scientific simulation and modelling drive the need for greater computing power.
- Single-core processors can not be made that have enough resource for the simulations needed.
  - Making processors with faster clock speeds is difficult due to cost and power/heat limitations
  - Expensive to put huge memory on a single processor
- Solution: parallel computing – divide up the work among numerous linked systems.

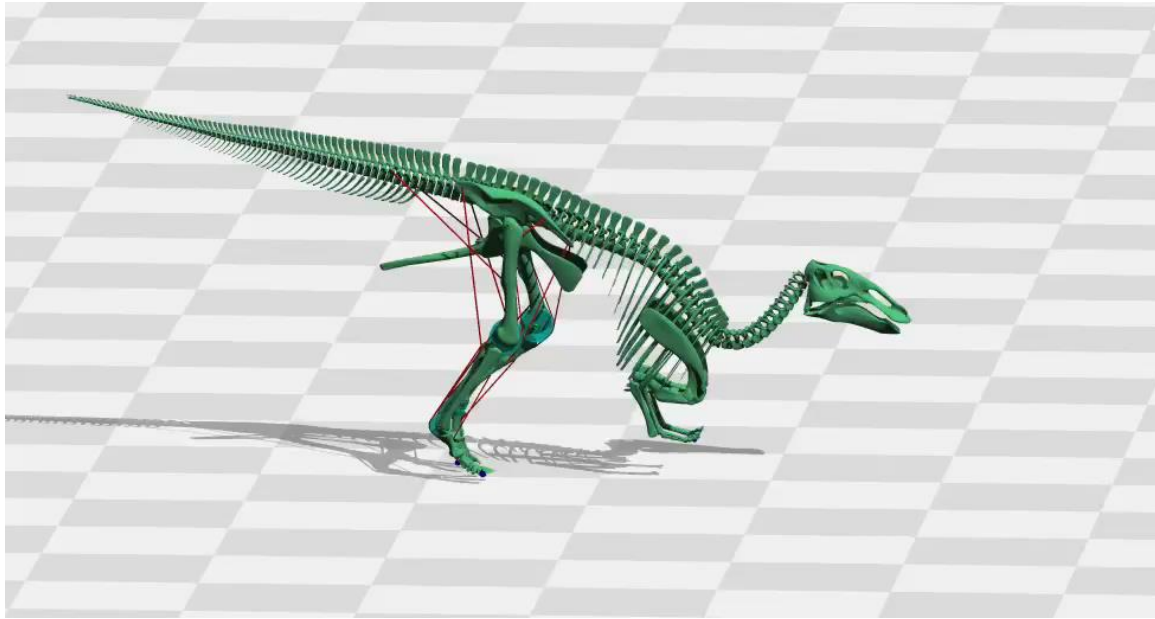


# Generic Parallel Machine

- Good conceptual model is collection of multicore laptops
  - come back to what “multicore” actually means later on ...
- Connected together by a network

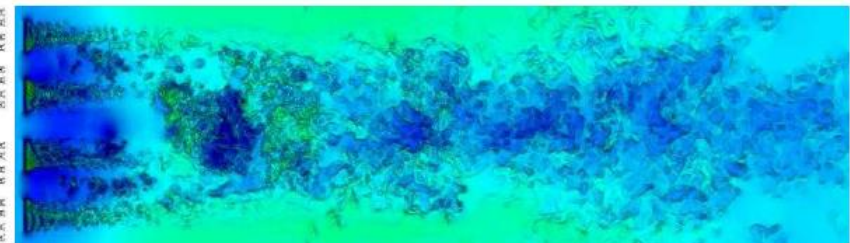
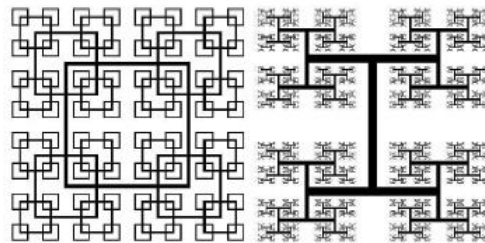
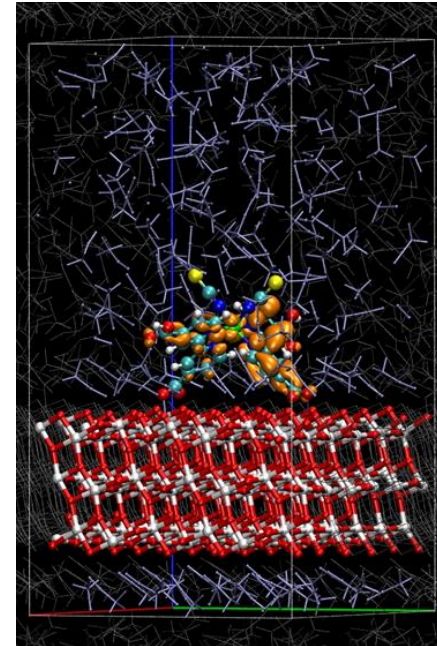


- Each laptop is called a *compute node*
  - each has its own operating system and network connection
- Suppose each node is a quadcore laptop
  - total system has 20 processor-cores



Modelling dinosaur gaits  
 Dr Bill Sellers, University of Manchester

Dye-sensitised solar cells  
 F. Schiffmann and J. VandeVondele  
 University of Zurich



Fractal-based models of turbulent flows  
 Christos Vassilicos & Sylvain Laizet,  
 Imperial College



# The Fundamentals

Why do I need to know this?



# Parallel Computing

- Parallel computing and HPC are intimately related
  - higher performance requires more processor-cores
- Understanding the different parallel programming models allows you to understand how to use HPC resources effectively





# Hardware Layout

- Understanding the different types of HPC hardware allows you to understand why some things are better on one resource than another
- Allows you to choose the appropriate resource for your application
- Allows you to understand the ways to parallelise your serial application
- Gives you an appreciation of the parts that are important for performance



# Differences from Desktop Computing

- Do not log on to compute nodes directly
  - submit jobs via a batch scheduling system
- Not a GUI-based environment
- Share the system with many users
- Resources more tightly monitored and controlled
  - disk quotas
  - CPU usage



# What do we mean by “performance”?

- For scientific and technical programming use FLOPS
  - Floating Point OPerations per Second
  - $1.324398404 + 3.6287414 = ?$
  - $2.365873534 * 2443.3147 = ?$
- Modern supercomputers measured in PFLOPS (PetaFLOPS)
  - Kilo, Mega, Giga, Tera, Peta, Exa =  $10^3$ ,  $10^6$ ,  $10^9$ ,  $10^{12}$ ,  $10^{15}$
- Other disciplines have their own performance measures
  - frames per second, database accesses per second, ...

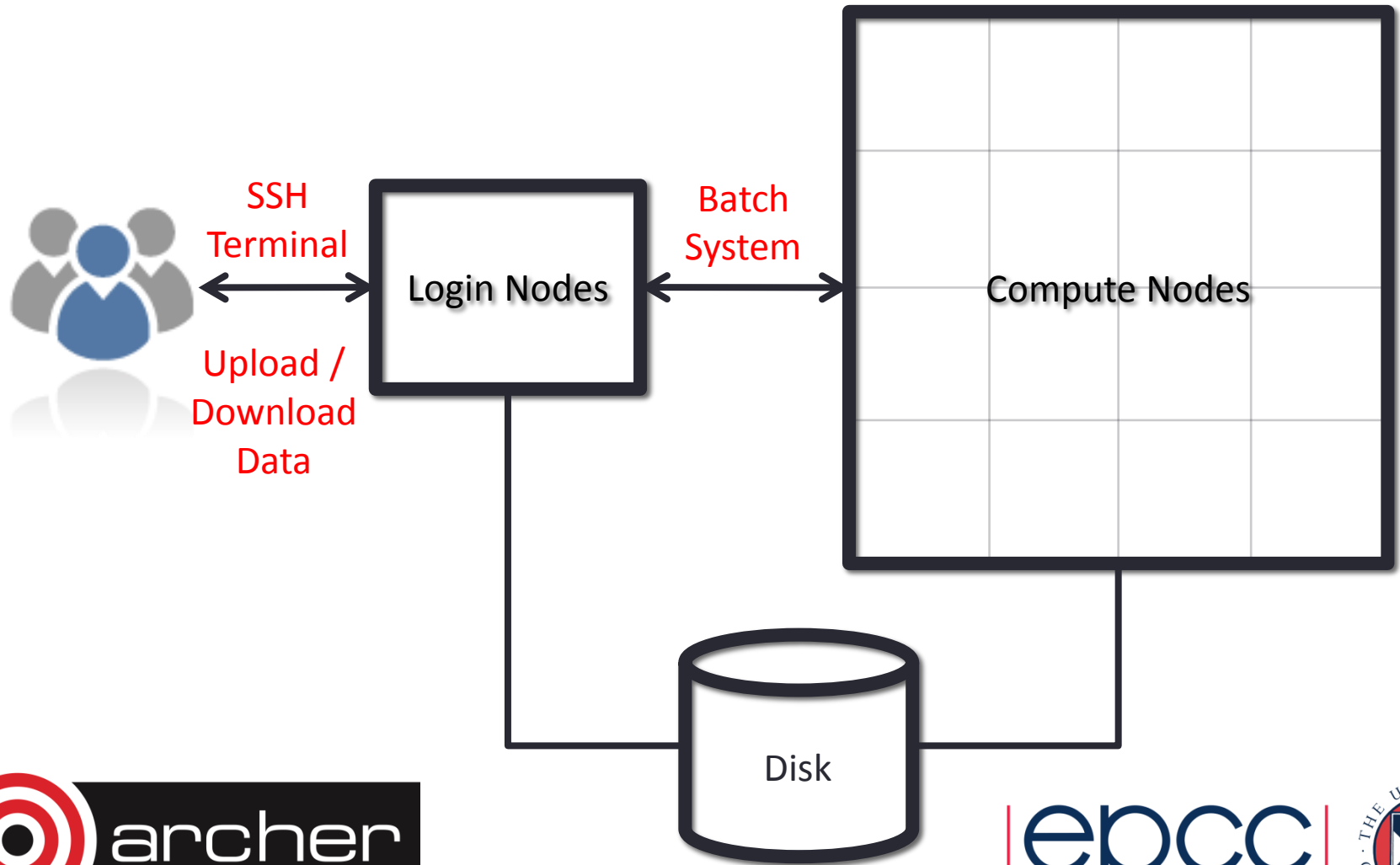


# HPC Layout and Use

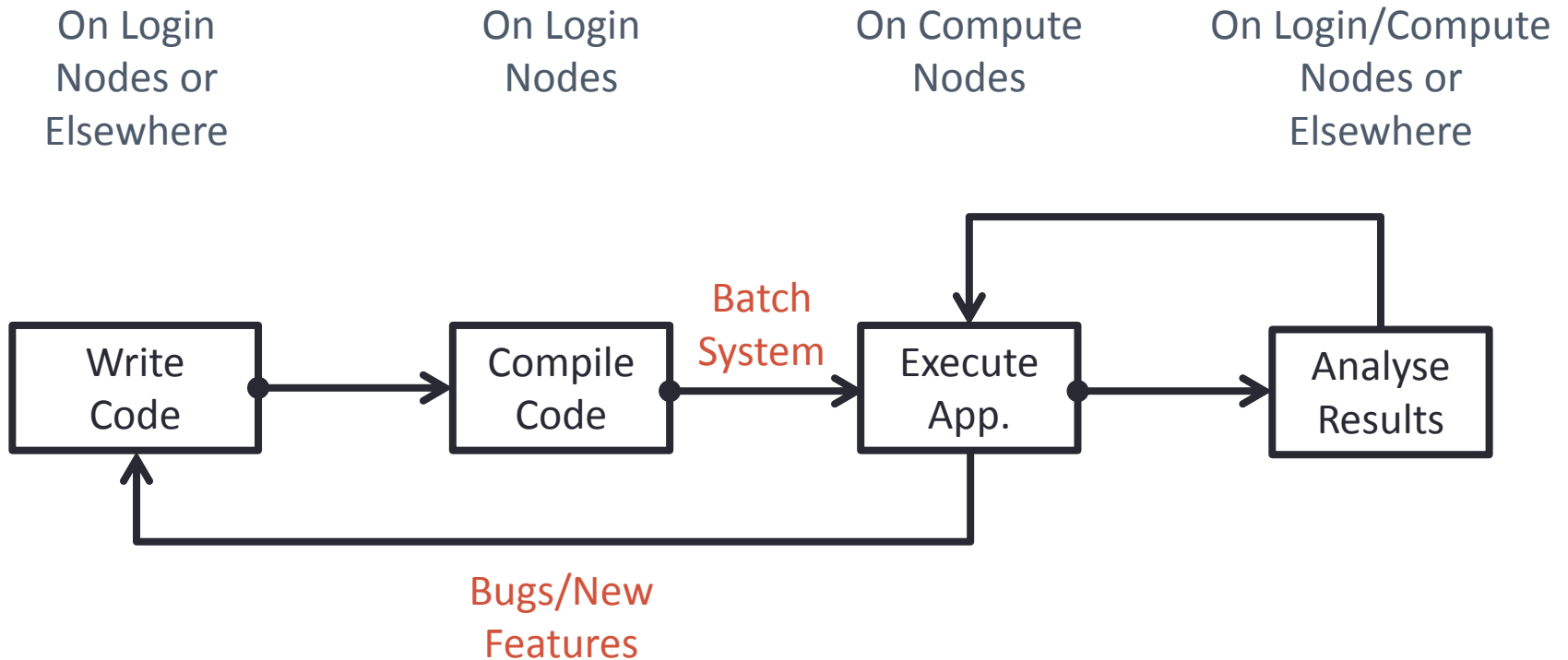
Starting concepts



# Typical HPC system layout



# Typical Software Usage Flow



# ARCHER



# ARCHER

- UK National Supercomputing Service
  - funded by EPSRC and NERC
  - operated by EPCC





# ARCHER in a nutshell

- Peak performance of 2.55 PFLOPS
- Cray XC30 Hardware
  - Intel Ivy Bridge processors: 64 (or 128) GB memory; 24 cores per node
  - 4920 nodes (118,080 cores) each running CNL (Compute Node Linux)
  - Linked by Cray Aries interconnect (dragonfly topology)
- Cray Application Development Environment
  - PBS batch system
  - Cray, Intel, GNU Compilers
  - Cray Parallel Libraries
  - DDT Debugger, Cray Performance Analysis Tools



# Summary

- High Performance Computing = parallel computing
- Run on multiple processor-cores at the same time
- Typically use fairly standard processors
  - but many thousands of them
- Fast network for inter-processor communications

