Level 7 Advanced Computational Physics – Mini Projects

Dr. Simon Hanna
Version 1.1

October 15, 2019

General Information

You will receive a list of titles for the Advanced Computing mini-project from the course coordinator (Dr Hanna). You will be required to produce:

1. A technical report describing the method implemented and results obtained;
2. A copy of your working program(s).

The technical report should be at a level understandable by final-year MSci physics undergraduates and have about 3000 words (excluding captions and references). Reports 15% longer or shorter will be penalized in the marking. They should be technical in style, including appropriate equations (with 'one equation = 1 word' on average), and cite relevant literature.

Mini projects will be marked according to the following criteria:

Presentation (Style, Appearance, Clarity, Fluency): This section concerns the presentation of both the report and the program. An excellent report will be easy and engaging to read, with the material laid out in a coherent, logical and accessible manner. It will have a professional appearance throughout with no errors in grammar or spelling. An excellent program will be logically structured and clearly laid out with appropriate indentation of program blocks and choice of variable names, and will be appropriately commented.

Scientific and Technical Content: This section assesses the scientific and technical quality and quantity of the program(s) and report; how well it demonstrates understanding and explains relevant concepts, the depth of conclusions, and the ability to bring together different areas. *An excellent report will include discussion of advantages of different parallelisation methods, shared memory versus distributed memory, system scaling (both model size and number of compute cores) and tuning.* It will include original explanations with evidence of knowledge of relevant literature.

Understanding, Criticism, Insight and Structure: This section assesses your understanding of the material presented in the report. An excellent report should demonstrate a thorough understanding of the material involved and include a critique of the parallel methods used and results achieved.

Mini-project topics

You may either choose one of the topics listed below, or propose a topic of your own. Further details on any of the topics, and general advice on choice of topic, are available from Dr Hanna. Please note that, in choosing a topic for your mini-project you should *avoid a direct overlap with your final year project*.

Many body problems

Molecular Dynamics: Write a parallel molecular dynamics code to simulate monatomic or diatomic gases. You should choose appropriate parameters for modelling, say, argon or nitrogen, and use your model to test basic principles of (non)ideal gases (see Leach, A.R., "Molecular modelling principles and applications", Longman, 1996).

Polymer Dynamics: A simple model for a polymer chain consists of a set of beads separated by springs; the beads interact through non-bonded interactions to prevent overlap, which the springs provide the expected entropic elasticity of the system. The motion of the beads can be modelled using either a Newtonian or Langevin (Brownian) dynamics. Write a parallel dynamics code to model polymer chains in the melt phase, and use it to test the expected static and dynamic polymer behaviour (Doi, M., "Introduction to Polymer Physics", Clarendon Press, 1996).

The Boid Model: The boid (bird-oid object) model for swarming and collective behaviour was developed by Craig Reynolds and published in 1987 (Reynolds, C., "Flocks, Herds, and Schools: A Distributed Behavioral Model", Computer Graphics **21**, 29-34, 1987). It is a Lagrangian algorithm, in which each individual particle obeys a set of rules. In particular: a boid wants to avoid colliding with others; a boid wants to match the velocity of nearby boids; a boid wants to stay close to nearby boids. These three rules were devised from the observation of how a flocking bird interacts with the other members of the flock, and in fact can be extended to any other swarm-like system. Write a parallel program to model the behaviour of a large swarm of boids.

Gravity Simulator (Direct Approach): Simulations of Newtonian gravity tend to be expensive, due to the long-range nature of the gravitational force. Write a parallel code, using a direct summation of all terms, and explore its use for simulating planetary systems with large numbers of planets and moons (Aarseth, S., "Gravitational N-Body Simulations: Tools and Algorithms", Cambridge University Press, 2003).

Gravity Simulator (Approximate Method): A number of approximate approaches have been developed to allow simulation of large gravitational systems, including trees (Barnes-Hut method), fast multipoles and particle-meshes. Write a parallel code exploiting one of the above techniques and use it to study the dynamics of cloud of interacting particles (Aarseth, S., "Gravitational N-Body Simulations: Tools and Algorithms", Cambridge University Press, 2003).

Monte Carlo methods

Ising Model: The Ising model is a mathematical model of ferromagnetism consisting of a lattice of magnetic dipole moments that can be in one of two states (+1 or -1). The spins interact with their neighbours and display phase transitions in 2-d and above. Write a parallel Ising code and use it to explore criticality as a function of model size for 2-d and 3-d systems (Brush, S.G., "History of the Lenz-Ising Model", *Rev. Mod. Phys.*, **39**, 883–893 (1967)).

Nanoparticle clusters: Colloidal systems consist of microscopic or nanoscopic particles in fluid, interacting via van der Waals potentials. Typically, colloids may either be stable (particles separated) or will coalesce (cluster formation). Write a parallel Monte Carlo program, using the Metropolis algorithm, to model the formation of nanoparticle clusters, and use it to explore colloid stability versus particle size and material (Metropolis,

N., Rosenbluth, A.W., Rosenbluth, M.N., Teller, A.H. and Teller, E., "Equation of State Calculations by Fast Computing Machines", *J. Chem. Phys.*, **21**, 1087–1092 (1953)).

Lattice-based simulations

Ising Model: See above.

Lattice Boltzmann flow simulator: Lattice Boltzmann Methods offer a way of computing a solution to a partial differential equation, based on dividing a spatial region into cells. The method works by considering densities of particles in cells, and fluxes of particles between cells. It is particularly successful for modelling flow in confined systems. Write a Lattice Boltzmann program to simulate the flow expected in microfluidic lab-on-a-chip device (Chen, S. and Doolen, G.D., "Lattice Boltzmann method for fluid flows", *Ann. Rev. Fluid Mech.*, **30**, 329–364, (1998)).

FDTD: The finite-difference time-domain model is used to model the propagation of electromagnetic waves through a medium. The method proceeds by numerically integrating Maxwell's curl equations on a lattice, leap-frogging the E and H fields at each time-step. Write a 2-d FDTD code to simulate the propagation of plane waves past various dielectric obstacles and use it to test your understanding of diffraction theory (Yee, K.S., "Numerical solutions of initial boundary value problems involving Maxwell's equations in isotropic media", *IEEE Trans. Ant. Prop.*, **AP-14**,302–307 (1966)).

Miscellany

Fast Fourier Transform: Fast Fourier transforms are required in many branches of physics, including molecular simulations and signal processing, and are an ideal candidate for parallelisation. Write a parallel FFT code, capable of working in 1, 2 and 3-dimensions and explore its efficiency as a function of sample size (Eijkhout, V., "Introduction to High Performance Computing", 2016).

Instructions for submission

Your report, all program and header files, together with appropriate make files or instructions for building the executable should be zipped together in a single archive and uploaded to Blackboard before the deadline in Week 15.