THE AUSTRALIAN NATIONAL UNIVERSITY
Mid Semester Examination – April 2010

COMP3320/6464
High Performance Scientific Computing

Study Period: 15 minutes
Time Allowed: 90 minutes
Permitted Materials: Non-Programmable Calculator
Questions are NOT equally weighted.

The questions are followed by labelled, framed blank panels into which your answers MUST be written. Extra boxes are given at the end of the paper. No additional paper will be provided. Writing outside of the boxes may not be marked.

This exam is marked out of 100. You should answer all questions.

Your mark for this exam will contribute 15% of your total course mark, according to the marking scheme given on the course web page.

Student Number:

The following are for use by your friendly examiner!

Q1 Mark Q2 Mark Q3 Mark Q4 Mark Total Mark
Question 1  [10 marks]  Computational Science Background

(a) For both: i) C; and ii) Python, give one positive and one negative relating to the use of that language for computational science applications.

(b) In the presentation by Dr John Taylor of CSIRO you heard how data plus algorithms plus visualization leads to new scientific discovery. Give one example of each of the following:  i) A scientific application where vast quantities of data must be processed with details of where the data comes from; ii) An algorithm that forms the basis of a widely used computationally intensive application (name or describe the algorithm briefly); iii) A scientific application where visualization is central to the presentation of the results and why.
Question 2  [30 marks] Floating Point Computation

(a) The following code illustrates the effects of both truncation and rounding errors:

```c
#include <stdio.h>
#include <math.h>
int main(void)
{
    float h,x;
    int i;
    x=12.3456789;
    for (i=0, h=0.1; i<14; i++, h*=0.1) printf("%16.14f %8.1e %16.14f \n",
            3.0*x*x,h,(((x+h)*(x+h)*(x+h))-(x*x*x))/h);
    return 0;
}
```

It computes the derivative of \(x^3\) both analytically and numerically using the finite difference scheme:

\[
\frac{dx^3}{dx} = 3x^2 \approx \frac{(x+h)^3 - x^3}{h}
\]

where \(h\) is the step size used in the finite difference. The code uses \(x = 12.3456789\) and various step sizes. It gives the following output:

- \(457.24739088661136 \ 1.0e-01 \ 460.96109472704148\)
- \(457.24739088661136 \ 1.0e-02 \ 457.61786125682272\)
- \(457.24739088661136 \ 1.0e-03 \ 457.28442892190822\)
- \(457.24739088661136 \ 1.0e-04 \ 457.2510946003245\)
- \(457.24739088661136 \ 1.0e-05 \ 457.2477612570375\)
- \(457.24739088661136 \ 1.0e-06 \ 457.24742793259342\)
- \(457.24739088661136 \ 1.0e-07 \ 457.24739445992828\)
- \(457.24739088661136 \ 1.0e-08 \ 457.2473912532003\)
- \(457.24739088661136 \ 1.0e-09 \ 457.2473909200175\)
- \(457.24739088661136 \ 1.0e-10 \ 457.2473906806185\)
- \(457.24739088661136 \ 1.0e-11 \ 457.24738147804351\)
- \(457.24739088661136 \ 1.0e-12 \ 457.24718045061832\)
- \(457.24739088661136 \ 1.0e-13 \ 457.2476124615535\)
- \(457.24739088661136 \ 1.0e-14 \ 457.24539944595568\)

(i) Define: i) Rounding error; ii) Truncation error.
(ii) Describe how this program illustrates both sources of error.

(iii) For any given value of \( h \), state how you would modify the code to: i) Decrease the effect of rounding errors; ii) Decrease the effect of truncation errors.

(iv) For \( h = 1.0e-07 \) write down an expression for the relative error of the numerical derivative. (You do not need to evaluate the expression.)
Question 3 [25 marks] Performance Analysis and Modelling

(a) For most applications it is best to have small number of very powerful processors rather than a larger number of less powerful processors: i) Explain why this is the case; ii) Suggest a class of applications for which this may not be true.

(b) Consider a multi-user computer system with a (minimum) time slice interval of \( t_s = 0.01 \) s. There is a moderate ‘process load’ on this system. A test program is to be used to time a computation; the program uses a high resolution wall timer (having overhead and resolution of much less than \( t_s \)).

Explain why you might want to: i) Ensure \( 0.5t_s \leq t_I < t_s \), where \( t_I \) is the timing interval of the computation; ii) Take several consecutive timings of the test program and then select the smallest time as the “true” value.
(c) When running your application program you have used hardware performance counters to measure instructions per cycle (IPC): i) On a single core of a modern hardware system what is the typical maximum value for IPC? ii) If your measured value is close to this maximum what does this tell you about the interaction of your program with the underlying hardware? iii) Why is it difficult to compare instruction counts obtained on fundamentally different computer architectures (eg. UltraSPARC with Pentium IV)?

[9 marks]
Question 4  [35 marks] High Performance Computer Architecture

(a) The typical memory hierarchy of a modern processor includes registers, cache, main memory and disk storage. Complete the empty cells in the following table with numerical values and units:

<table>
<thead>
<tr>
<th>Name</th>
<th>Typical Size</th>
<th>Access Time</th>
<th>Bandwidth</th>
<th>Managed by</th>
</tr>
</thead>
<tbody>
<tr>
<td>Registers</td>
<td></td>
<td>1 cycle</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Cache</td>
<td></td>
<td></td>
<td>~100GB/s (Level 1)</td>
<td></td>
</tr>
<tr>
<td>Main memory</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Disk</td>
<td>0.5-2TB</td>
<td></td>
<td></td>
<td>OS</td>
</tr>
</tbody>
</table>

[12 marks]

(b) Modern processes exploit cache systems; i) What is a cache line conflict? ii) Provide a small code fragment that will exhibit a cache line conflict if executed on a computer with a 16KB 2-way set associative cache that has a 32byte cache line size.

[8 marks]
(c) Over time there has been a tendency for all microprocessors to increase the number of stages in their pipeline: i) Give one advantage of long pipelines; ii) Give one (fundamentally different) disadvantage of long pipelines; iii) State how microprocessors designers have sought to minimize the disadvantage of long pipelines.

(d) The following shows two lines of C code that are executed in a loop defined by \texttt{for (i=0; i<n; i++)}. For each line of C code detail the number of operations you expect to be performed per loop iteration of the following type; i) load, ii) store, iii) floating point. Explain how you derive your results.

A) \( a = a + x[i]*x[i]; \)
B) \( x[i] = (a*y[i]+b*y[i])*c; \)