UNIVERSITY OF BIRMINGHAM

School of Computer Science

Third Year - BSc Artificial Intelligence and Computer Science
Third Year - BSc Computer Science
Third Year - MSci Computer Science
Fourth Year - BSc Computer Science w Study Abroad
Third Year - MSci Mathematics + Computer Science w Industrial Year
Fourth Year - BSc Computer Science w Industrial Year
Third Year - MEng Comp Science/Software Engineering w Industrial Year
Fourth Year - BSc Artificial Intelligence and Computer Science w Industrial Year
Third Year - MSci Computer Science w Industrial Year

06 26945

Distributed and Parallel Computing

Main May/June Examinations 2018

Time allowed: 01:30

[Answer ALL Questions]

1. This question is about GPU programming and the CUDA programming model.

   (a) Amdahl's law and the Gustafson-Barsis law provide different insights into opportunities for improved computational performance through parallelisation. They also clarify limits on such improvements. Explain what their lessons are with particular reference to the kinds of problems that can benefit from parallelisation in different ways. **[10%]**

   (b) Given a large array of numbers X, and a predicate $p()$ that evaluates a boolean condition on these numbers, a compact array of only the numbers that satisfy the predicate is required.

   Assuming that primitive kernels to do map, reduce, scan (inclusive and exclusive, normal and segmented), scatter and gather are available, explain how an efficient parallel version of the required operation can be implemented.

   Finally, use a diagram to show the results of each step on an array of the 12 integers 1 to 12, where the predicate is p(x): x $\rightarrow$ ((x % 3) == 0) **[20%]**

(c) A student wants to write an efficient CUDA program to transpose a square matrix. The kernel she writes and the code to invoke it are as follows:

```
#define N 1024
#define K 32
__global__ void transpose(const int in[], int out[])
{
    int in_c_i = blockIdx.x*K;  int in_c_j = blockIdx.y*K;
    int out_c_i = blockIdx.y*K; int out_c_j = blockIdx.x*K;
    int x = threadIdx.x;        int y = threadIdx.y;

    __shared__ int tile[K][K];

    tile[x][y] = in[(in_c_i + y) + (in_c_j + x) * N];
    __syncthreads();
    out[(out_c_i + x) + (out_c_j + y) * N] = tile[x][y];
}
...
    dim3 blocks(N / K, N / K);  dim3 threads(K, K);
    transpose<<<blocks, threads>>>(d_in, d_out);
```

The program calculates the correct results, but with disappointing performance. The student profiles the kernel and obtains the following results:

Factor	GTX 1080 Ti
Duration:	58.215μs
Register/Thread:	8
Shared Memory/Block:	4 KiB
Global Load Efficiency:	12.5%
Global Store Efficiency:	100%
Shared Efficiency:	3.1%
Warp Execution Efficiency:	100%
Non-Predicated Warp Execution Efficiency:	100%
Occupancy Achieved:	89.4%
Occupancy Theoretical:	100%
Occupancy Limiter:	none

(i) For each item (other than the obvious *duration*) in the profile report, explain how the student should interpret it, and estimate its relevance to the problems with this kernel's performance as one of *"very relevant"*, *"mildly relevant"* or *"irrelevant"*. **[10%]**

(ii) Explain the the sources of the performance problems in the student's code as indicated by the profiling report. **[10%]**

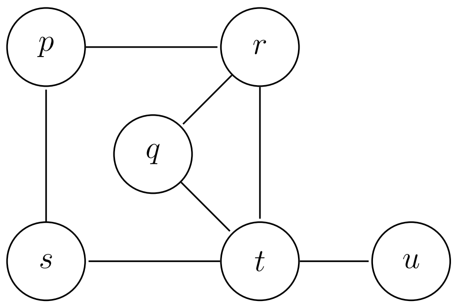
(iii) Describe the changes necessary in the code to correct the problems and explain why they work. **[10%]**

2. This question is about distributed algorithms

   (a) Draw the time-space diagram for a simplest possible example (i.e. smallest in number of events) that includes concurrent events a and b where $\mathrm{LC}(a) < \mathrm{LC}(b)$ for a Lamport Clock LC and indicate the clock values on the events in the diagram. Show also the vector clock values for the same events and show that $\mathrm{VC}(a)$ and $\mathrm{VC}(b)$ are incomparable. **[10%]**

   (b) Design a strategy by which Logical Clocks can be used to implement a distributed mutual exclusion algorithm assuming a network of reliable FIFO channels. **[10%]**

   (c) Consider the following undirected FIFO network:



   Taking p as the initiator, demonstrate a sample execution of the Tarry traversal algorithm by annotating the channel lines as follows:

   - Add message numbers to show the order in which the messages are sent. Any total ordering compatible with the algorithm is acceptable.
   - Add arrows with the message numbers to show the direction in which the messages are sent.
   - Indicate the resulting spanning tree by adding arrowheads to the corresponding channel lines to show the parent relation.

   [10%]

   (d) Prove that the rules of the the Tarry traversal algorithm ensure that the token finally ends up at the initiator. **[10%]**

Do not complete the attendance slip, fill in the front of the answer book or turn over the question paper until you are told to do so

Important Reminders

- Coats/outwear should be placed in the designated area.
- Unauthorised materials (e.g. notes or Tippex) must be placed in the designated area.
- Check that you do not have any unauthorised materials with you (e.g. in your pockets, pencil case).
- Mobile phones and smart watches **must** be switched off and placed in the designated area or under your desk. They must not be left on your person or in your pockets.
- You are not permitted to use a mobile phone as a clock. If you have difficulty seeing a clock, please alert an Invigilator.
- You are not permitted to have writing on your hand, arm or other body part.
- Check that you do not have writing on your hand, arm or other body part – if you do, you must inform an Invigilator immediately
- Alert an Invigilator immediately if you find any unauthorised item upon you during the examination.

Any students found with non-permitted items upon their person during the examination, or who fail to comply with Examination rules may be subject to Student Conduct procedures.