Concurency and prallel programming

Final written exam

Date: December, 21 2017
Time: 14:00-16:00

Number of pages: 7 (including front page)
Number of questions: 25
The exam counts for: 25% of the final CPP grade
NOTE: Questions might have more than one correct answer

BEFOR YOU START

- Wait until you are instructed to open the exam sheet
- Check if your version of the exam is **complete**
- Write down **your name**, **student ID number** and if applicable the **version number** of the exam, on **each sheet** that you hand in. **Also number the pages**
- Your mobile phone has to be switched off and put in your coat or bag, Your coat and bag should be under you table
- Tools allowed: rule, (graphic) calculator. ~~Books~~, ~~articles~~, ~~tables~~, none, etc

PRACTICAL MATTERS

- The first 30 minutes and the last 15 minutes you are not allowed to leave the room, not even to visit the toilet
- You are obliged to identify yourself at the request of the examiner (or a representative) with proof of your enrollment or a valid ID
- 15 minutes before the end of the exam you will be warned that the time to hand in is approaching
- During the examination it is not permitted to visit the toilet unless the invigilator gives permission to do so
- If applicable please fill out the evaluation of the course

A. Computer Architecture

1. What is the consequence of the end of Dennard scaling?
   a. The number of transistors that can be integrated on a chip will not increase anymore
   b. The frequency scaling of microprocessors (i.e. the increase of clock frequencies) has stopped
   c. The area of chips will not decrease anymore with any new generation of (smaller) transistors
   d. There is no improvement of transistor speed anymore when reducing the size of transistors

2. Assume a 100-core multi-processor system with an application of which 90% of the code can be parallelized. What is the (rounded) maximum speedup according to Amdahl's law?
   a. 4x
   b. 9x
   c. 20x
   d. 90x

B. Parallel Computer Architectures: Multi-core processors / Multicore Scaling / Memory organization / Communication Architectures

3. Select the true statements:
   a. VLIW processors allow for parallel execution of instructions.
   b. VLIW processors rely on the compiler to identify parallelism.
   c. VLIW processors have problems in case of non-deterministic behavior of instructions.
   d. VLIW processors perform instruction scheduling in hardware.

4. Select the true statements:
   a. SIMD ISA extensions amortize cost/complexity of managing an instruction stream across many ALUs.
   b. Caches reduce the average memory latency.
   c. Multi-threading does not require the replication of the register file.
   d. Hardware-supported simultaneous multi-threading requires the compiler to choose instructions from multiple threads.

5. A processor running at 1GHz, featuring 2MB of on-chip cache and 128 arithmetic units computes element-wise multiplication of two vectors A and B and stores the results in vector C. Vectors A, B, and C are stored in off-chip DRAM and contain IEEE double precision floating point data. Assume that the vectors contain millions of elements.

For each element with index i:

1. Load input A[i]
2. Load input B[i]
3. Compute A[i] x B[i]
4. Store result into C[i]

Select the true statements

a. The memory operations amount to 12 bytes for every multiplication=
b. Doubling the cache size to 4MB reduces the off-chip memory bandwidth required to keep the arithmetic units by 50%.
c. The memory bandwidth required to keep the arithmetic units busy is ~3TB/s.
d. The memory bandwidth required to keep the arithmetic units busy is~1.5TB/s

6. Select the true statements:
   a. In NUMA designs, all processors can access any memory location
   b. In NUMA designs, the cost of memory access is the same for all processor.
   c. In NUMA designs, the cost of memory access is different for different processors.
   d. NUMA designs provide higher bandwidth to local memory.

C. OpenMP/PThead/MPI

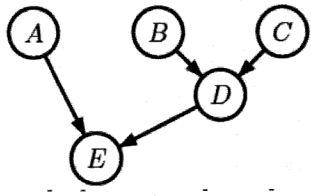
7. Select the true statements:
   a. OpenMP uses `#pragma omp parallel` to generate a parallel section.
   b. OpenMP `#pragma omp for` to generate a parallel section.
   c. OpenMP code using `#pragma omp parallel for` does not compile if the targeted for loop has dependencies.
   d. OpenMP code using `#pragma omp parallel for` might not execute correctly if the targeted for loop has dependencies.

8. Take the following C+OpenMP code snippet executing on a machine with at least 5 threads:

```
1. int sum=0;
2. omp_set_num_threads(5); // threads 0-4
3. #pragma omp parallel for schedule(static,1) shared(sum)
   private(s)
4. for (i=0; i<10; i++) {
5.      sum+=i;
6.      s+=i;
7. }
8. printf("sum = %d \n", sum);
```

   a. At the end of the loop, sum=45.
   b. To ensure sum=45, the shared(sum) clause should be replaced with a reduction(sum:+)
   c. Just before completion, for thread 0, s is 5.
   d. Adding a nowait clause to the pragma will not change the result of the code.

9. Take the following dependency graph between functions A,B,C,D,E, each executed by 1 thread. Considering only pthread_create and pthread_join (in short: c() and j()), which of the following codes generate this graph with maximum parallelism:



a. c(A),c(B),c(C),c(D),c(E),j(B),j(C),j(A),j(D),j(E)
b. c(B),c(C),j(B),j(C),c(D),j(D),c(A),j(A),c(E),j(E)
c. c(A),j(A),c(B),j(B),c(C),j(C),c(D),j(D),c(E),j(E)
d. c(A),c(B),c(C),j(B),j(C),c(D),j(D),j(A),c(E),j(E)

10. Two threads, A and B, work in parallel. x and y are shared variables. m is a mutex lock:

`A: read x;`
`lock(m);`
`y = y+x;`
`write(y);`
`unlock(m)`
	`B: read y;`
	`lock(m);`
	`x = x+y;`
	`write(x);`
	`unlock(m);`

a. There are no race conditions in this code.
b. There is no deadlock between A and B.
c. The lock(m) statements are incorrectly placed to protect for race conditions.
d. With or without the lock/unlock statements, the code works the same.

11. To be able to enable overlapping of communication with computation
a. You have to use synchronous communication routines.
b. You have to use asynchronous communication routines.
c. You have to use synchronization routines.
d. You have to use point to point communications.

12. Order of MPI messages call is guaranteed when
a. Two processes in a single communicator (one source-one destination-one communicator)
b. Three processes in a single communicator (two sources-one destination-one communicator)
c. Two processes in a two different communicators (one sources-one destination-one two communicators)
d. Two processes in a two different communicators (two sources-one destination-one two communicators)

D. Big Data platforms (MapReduce/Spark)

13. In Spark Resilient Distributed Datasets refer to
    a. Immutable, Partitioned collections of Data records created through ACTIONS.
    b. Immutable, Partitioned collections of Data records created through TRANSFORMATION.
    c. Immutable, Partitioned collections of Data records created through LINEAGE.
    d. Immutable, Partitioned collections of Data records created through CACHING.

14. In Hadoop Distributed File Systems, which of the following sequence is correct to check whether or not the data nodes are ready to store data blocks:
    a. (1) The NameNode sends the list of data nodes to the client, (2) the client contact the 1st node on the list, (3) 1st node contact the 2nd node on the list etc.
    b. (1) The NameNode sends a message to the data nodes to get ready to receive the data, (2) the data nodes send the acknowledgement to the NameNode, (3) the NameNode sends the list of available data nodes to the client.
    c. (1) The client sends the data blocks to the NameNode, (2) The NameNode sends a message to data nodes to get ready to receive the data, (3) the data nodes send the acknowledgement to the NameNode.
    d. (1) The client sends the data blocks to the NameNode, (2) The NameNode sends the data blocks to data nodes, (3) the data nodes send the acknowledgement to the NameNode that they have received the data block.

E. Performance and GPU programing (CUDA)

15. Take the following code fragment (assuming X is a global variable) which counts the number of positive values in an array using CUDA. The code is launched for N elements, N/1024 blocks, and 1024 threads per block.

```
1. __global__ myCounter(int *myArray) {
2.   myID = myThreadId(); //this returns the global threadID
3.   if (myArray[myID] > 0) X = X+1;
4. }
```

Select only the true statements:
    a. This code never works correctly.
    b. To work correctly, the code needs an atomic operation for incrementing X.
    c. This is not a kernel.
    d. This kernel doesn't work correctly for N=1.000.000;

16. Take the following kernel, where the size of the array is 1M-elements and doAction1() and doAction2() take the same amount of operations (i.e., have the same performance):

```
1. __global__ doSomething(int *myArray) {
2. myID = myThreadId(); //this returns the global threadID
3. if (myArray[myID] > 0)
4.     doAction1();
5. else
6.     doAction2();
7. }
```

Select only the true statements:

a. The performance of the kernel is data independent.
b. The kernel's best performance is for an array with all numbers positive.
c. The kernel's best performance is for an equal number of positive and negative elements.
d. The kernel's worst performance is when there's at least one negative and one positive number in every 32 array elements.

17. Select the true statements only. Shared memory on NVIDIA GPUs is ...

a. Shared by all the cores.
b. Accessible to all the threads in the grid.
c. Used to implement software caching.
d. Difficult to use due to its coalescing requirements.

18. Thread synchronization on GPUs. Select the true statements only.

a. The barrier functionality cannot be achieved on GPUs.
b. The barrier functionality on GPU thread-blocks can be realized using a built-in statement.
c. One can use atomic operations to synchronize access to global variables only between threads in the same block.
d. Because shared memory is only accessible to one block, there are no race conditions in shared memory.

19. Assume an NVIDIA GPU with 10 SMs, each one with 32 cores, running at 0.5GHz. What is the peak performance of this GPU.

a. 100 GFLOPs
b. 200 GFLOPs
c. 320 GFLOPs
d. 160 GFLOPs

20. Assume the following code, which transposes a matrix.

```
1. for (i=0; i<N; i++)
2.    for (j=0; j<N; j++)
3.      B[j][i] = A[i][j];
```

Matrices contain integer numbers, with 1 int = 4 Bytes. The code runs on a machine with 1TFLOP theoretical peak and 200GB/s peak bandwidth. Select the true statements:

a. The arithmetic intensity/operational intensity of the kernel is 0.
b. The arithmetic intensity of the kernel is 0.25.
c. On the given machine, this is a memory-bound application.
d. If the code executes in about 4ms, N is around 10000.

21. Take an application A solving a problem of size N, and a machine M with c cores. We note T(A(N),M(c)) the execution time of A on machine M.

a. Computing the speedup of T(A(N),M(c1)) versus T(A(N),M(c2)) is a strong scaling analysis.
b. Computing the speedup of T(A(N1),M(c)) versus T(A(N2),M(c)) is a strong scaling analysis.
c. If T(A(N),M(1)) / T(A(N),M(c)) is larger than c, we talk about superlinear speedup.
d. T(A(N),M(1)) / c * T(A(N),M(c)) is a measure of efficiency.

F. Other Topics (SOA, Grid, Cloud, Guest lecture)

22. Client servers architecture requires that:

a. The client is aware of implementation details of the server.
b. The client knows how to invoke a server.
c. The client and the server are implemented in the same programming language.
d. The client and the server are implemented by the same programmer.

23. In the client server architecture which variant implies more processing on the clients side:

a. File Server Architecture
b. Database Server Architecture
c. Three-tier Architecture
d. Service Oriented Architecture

24. In the service oriented model the broker helps:

a. The requestor to identify available service providers.
b. The requestor to discover the interface of a given service provider.
c. To load balance the requests to different service providers.
d. The service providers to serve more than one requestors.

25. REST is:

a. *An architecture style* for designing networked applications.
b. A W3C standard framework for creating statefull services .
c. An XML vocabulary that provides a standard way of describing service Interfaces.
d. An XML-based lightweight protocol for the exchange of information in a decentralized way.