Knowledge Survey

This is a so-called **knowledge survey, not a test**. That means that you do not really answer the questions provided, but state your **confidence** to answer the question with your **present** knowledge. There will be very few exceptions that are explained below.

Purpose

The purpose of this survey is to provide participants with concepts that will be covered throughout the course and that enable participants to monitor their own progress. Especially, participants will be asked to fill out the same survey at the end of the course again.
Most important, these pre- and post-surveys are part of a research project that investigates the parallel programming productivity in high-performance computing and that looks into the impact of pre-knowledge & learning progress on programming productivity. That is also the reason that few questions are outside of the course content.

Since recent work showed that there is a tendency to not correctly predict one's confidence if not really answering the question, we ask participants to really answer **six** of the questions at the beginning of the survey. This shall help you to judge the complexity of the different types of questions (according to *Bloom's Level: knowledge, comprehension, application, analysis, synthesis, evaluation*).

Instructions

For each item in the survey, rate (on a three-point scale) your confidence to answer the question with your present knowledge. Please fill in A, B or C according to the explanations below.
For the first six questions, **first answer** the question to the best of your present ability and **second rate your confidence** in your answer.

- Mark an "**A**" as response to the question if you feel confident that you can now answer the question sufficiently for graded test purposes.
- Mark a "**B**" as response to the question if you can now answer at least 50% of it or if you know precisely where you could quickly get the information needed and could return here in 20 minutes or less to provide a complete answer for graded test purposes.
- Mark a "**C**" as response to the question if you are not confident that you could adequately answer the question for graded test purposes at this time.

Do your best to provide a totally honest assessment of your present knowledge. When you mark an "A" or a "B," this states that you have significant background to address an item.

Your responses will be used only for the purposes given above. However, we need to map pre- and post-surveys. Thus, please use the same name/ identifier on both knowledge surveys. Please, also denote the same name/ identifier on your (electronic) development diaries, student projects and other questionnaires. Your responses will be identifiable only to the instructor and research assistants of the course and kept private. All published data will be anonymized.

Name or other identifier: Nathan Seiger

Completion date: 2/25/2016

Overview

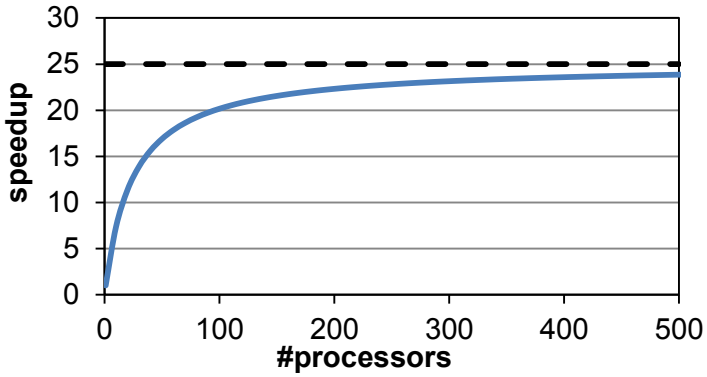
Please really answer these 6 questions (one question per page) and state your confidence.

1. Explain the principle of *prefetching* (if you want, you can also visualize it). Additionally explain, why prefetching is applied.
    a. Please put your answer of the question below - to the best of your present knowledge.

Prefetching is the action of pulling data from memory that is located near the data being accessed. This improves efficiency be allowing since most data is stored sequentially and most data structures and loops access data in order.

    b. Rate your confidence in your answer to this question.
    ☐ A
    ☑ B
    ☐ C

2. The following figure represents the speedup S_p behavior of a parallel program applying Amdahl's Law. Read the needed values from the figure and compute the parallel portion *p* of the program.



a. Please put your answer of the question below - to the best of your present knowledge.

Speedup = 1/(1-parallelPortion) + (parallelPortion / numCpus)
20 = 1/((1-pP)+(Pp/100))
20 = (100/(100-99pP))
5=100-99pP
pP=95/99
pP ~ .95

Honesty, had to remind myself Amdahl's Law formula

b. Rate your confidence in your answer to this question.
- ☐ A
- ☑ B
- ☐ C

3. Algorithms theory allows us to place an upper bound on the execution time of an abstracted algorithm on an abstract machine. How do you put an upper bound on performance of a specific, instantiated algorithm running on an actual machine? Explain this by giving a small code example on a self-chosen machine.
    a. Please put your answer of the question below - to the best of your present knowledge.

Placing an upper bound on an abstracted algorithm is done by performing analysis of the structures within the algorithm and the theoretical execution time required to perform the instructions in the algorithm. To determine an upper bound of an algorithms implementation the architecture of the machine must be considered. I am not sure what exactly the question is asking for so far as the code requirement. Things that would affect the run time would be system architecture such as word size, pipeline efficiency, whether or not the branch predictor is efficient and the code is optimized for the branch predictor of the machine in question. Other considerations are cache size, number of parallel cpu's, how memory is used in the algorithm, arrays/pointers/etc.

    b. Rate your confidence in your answer to this question.
        ☐ A
        ☑ B
        ☐ C

4. Execution of the following code with a single thread produces the result c=7. When executing the code with two threads, the result is also c=7. True or false? Justify your answer.

```
int a = 0, b = 0, c = 0;
#pragma omp parallel
{
      #pragma omp single
      {
            a += 3;
            #pragma omp barrier
      }
      #pragma omp single
      {
            b += 4;
            #pragma omp barrier
      }
      c = a + b;
}
```

a. Please put your answer of the question below - to the best of your present knowledge.

I had to look up pragma single and barrier. I now understand that these calls allow/deny threads access to a code block. If the barrier command was not included, the code result would be the same. Since there is a call to single, which allows only one thread access the the block, and there is a barrier, which requires that all threads meet before continuing execution, I think the code would hang indefinitely.

b. Rate your confidence in your answer to this question.

☐ A
☑ B
☐ C

5. Design a simple SPMD program that contains a deadlock. You can use MPI function names or pseudo code. For simplicity, assume that your code is always run with two processes and that all point-to-point operations are blocking operations.
   Please put your answer of the question below - to the best of your present knowledge.

```
#include<mutex>
#include<thread>

int main()
{
mutex m;
 int a=1;
 int b=2;
 int c=0;
thread t1;
thread t2;

for(int i = 0; i < 10; i++)
{
m.lock();
if(b < 2)
{
m.unlock();
}
a = a + b;
}
m.lock();
//hang here since b is always 2
```

a. Rate your confidence in your answer to this question.
   - ☐ A
   - ☑ B
   - ☐ C

6. Analyze the following code that implements a 2D stencil Jacobi solver. Given is the double-precision array $A \in R^{(imax+1)\times(kmax+1)\times 2}$ where *imax* and *kmax* are big integer values. Evaluate the general throughput of this code (without computing the exact numbers).
   Assume that the application shall run with a single thread on an x86 system that includes 4 cores, 256 Byte L1 cache (cache line size of 32 Byte) and 24 GByte of main memory. If the throughput performance is improvable with the given assumptions, how would the optimization look like (explain the tuning concept) and why?

```
int t0=0, t1=1, tmp=0;
for (int it=0; it<itmax; ++it) { // suitable no of iterations
      for (int i=1; i<imax; ++i) {
            for (int k=0; k<kmax; ++k) {
                  A[i][k][t1] = (A[i+1][k][t0] + A[i-1][k][t0] +
                                 A[i][k+1][t0] + A[i][k-1][t0]) * 0.25;
            }
      }
      // swap arrays
      tmp = t0; t0=t1; t1=tmp;
}
```

   a. Please put your answer of the question below - to the best of your present knowledge.

Instead of swapping the arrays, why not have two statements inside the innermost loop that perform the swap, would this cut the time by somewhere between 30-40%?

   b. Rate your confidence in your answer to this question.
      ☐ A
      ☐ B
      ☑ C

Computer Architecture

7. Visualize schematically a *2-socket multi-core NUMA* computer. The sketch should contain: multiple cores, L1/L2/L3 caches, main memory and the data links between caches and main memory.
    a. Rate your confidence to answer this question with your present knowledge.
        ☑ A
        ☐ B
        ☐ C

8. Determine the cache hit rate for the code given below. Assume that the size of the cache lines is 64 Bytes and that one double value has a size of 64-bit. Further assume that the compiler does not apply any optimizations (e.g. like prefetching), the variable *tmp* is hold in a register and the accesses to *i* must not be counted.

```
double a[800];
double tmp = 0;
/* .. */
for (int i=0; i<800; ++i) {
      tmp += a[i];
}
```

    a. Rate your confidence to answer this question with your present knowledge.
        ☐ A
        ☑ B
        ☐ C

9. Be there a cache of access time T_c = 8 ms and a local memory with access time T_m = 24 ms. Calculate the average access time T_{av} for an arbitrary memory location for the previous example. Additionally, calculate how much speedup is gained due to the use of the cache. Hint: Speedup S = access to main memory in relation to the average access time
    a. Rate your confidence to answer this question with your present knowledge.
        ☐ A
        ☑ B
        ☐ C

10. Apply the *unroll* technique to the following code to enable vectorization. The code uses arrays with a size of *N*. Use an unroll factor of 3 and assume that *N* is always divisible by 3 without remainder. Assume that there is no pointer aliasing.

```
#define N // with a big number that is divisible by 3
/* .. */
double r[N];
double q[N];
/* .. */
for (int i=0; i<N; ++i) {
      q[i] = r[i] + i;
}
```

    a. Rate your confidence to answer this question with your present knowledge.
        ☐ A
        ☐ B
        ☑ C

11. Assume a computer architecture with three level of caches and main memory: L1 16KB, L2 64KB, L3 8MB, main memory 4GB. Draw schematically the memory bandwidth of this architecture as function of the data size (double-precision elements). Mark the different cache levels/ main memory in your figure.
    a. Rate your confidence to answer this question with your present knowledge.
        ☐ A
        ☑ B
        ☐ C

12. Given is the data structure *mydata* in a C program. If the data access pattern is given as below, an efficient vectorization with 128-bit vector registers will be applied by the compiler. True or false? Justify your answer.

```
struct mydata {
      float myFloat;
      double myDouble;
};
struct mydata *array; // of size n
//…
// data access pattern:
for (int i=0; i<n; i++) {
      x += array[i].myFloat + array[i].myDouble;
}
```

    a. Rate your confidence to answer this question with your present knowledge.
        ☑ A
        ☐ B
        ☐ C

Tuning & Parallelization

13. Describe the theory of roofline analysis and how it can be used to distinguish hardware and software bottlenecks.
    a. Rate your confidence to answer this question with your present knowledge.
        ☐ A
        ☐ B
        ☑ C

14. From a programmer's point of view, describe the most important differences of shared-memory and distributed-memory programming. Name an example for each parallel paradigm.
    a. Rate your confidence to answer this question with your present knowledge.
        ☑ A
        ☐ B
        ☐ C

15. From an experiment, we know that for a given problem an application runs for 18 minutes with one thread and 4 minutes with 8 threads. What speedup does the parallel execution reach? What parallel efficiency does the parallel execution achieve? Based on Amdahl's law, how long does the application run in parallel and how long in serial?

a. Rate your confidence to answer this question with your present knowledge.
   ☐ A
   ☑ B
   ☐ C

16. We can derive an upper performance limit for parallel applications applying Amdahl's Law. If we cannot get more than this speedup with an infinite number of cores, why do we have big HPC systems with lots of parallel hardware at all?
    a. Rate your confidence to answer this question with your present knowledge.
       ☑ A
       ☐ B
       ☐ C

OpenMP

17. Give three different possible outputs of the following code running with three threads.

```
int main(int argc, char* argv[]) {
      #pragma omp parallel
      {
            int ID = omp_get_thread_num();
            printf("Hello World!");
            printf("I am thread %d.", ID);
      }
}
```

    a. Rate your confidence to answer this question with your present knowledge.
       ☑ A
       ☐ B
       ☐ C

18. Explain what a *static* and *dynamic* loop schedule means in OpenMP. Briefly list one typical use case for each schedule.
    a. Rate your confidence to answer this question with your present knowledge.
       ☐ A
       ☐ B
       ☑ C

19. If we set *T=4* and *N=400*, what is the final content of *A[0],..,A[T-1]* of the code below?

```
int A[T]; // A initialized with zeros
#pragma omp parallel num_threads(T)
{
      int ID = omp_get_thread_num();
      #pragma omp for schedule(static)
      for (int i=0; i<N; i++) {
            A[ID]++;
      }
}
```

    a. Rate your confidence to answer this question with your present knowledge.
       ☐ A
       ☑ B
       ☐ C

20. What is the output of this program if it is running with a single thread? What is the output if it runs with six threads?

```
int result = 0;
#pragma omp parallel
{
      int ID = omp_get_thread_num();
      #pragma omp critical
      {
            result += ID;
      }
}
printf("%d", result);
```

a. Rate your confidence to answer this question with your present knowledge.
   - ☐ A
   - ☐ B
   - ☑ C

21. Parallelize the following code applying the OpenMP tasking concept. If two threads are started, each thread should execute one of the tasks and they should run simultaneously.

```
int main(int argc, char* argv[]) {
      int result, result1, result2;
      #pragma omp parallel
      {
            #pragma omp single
            {
                  result1 = task1(42);
                  result2 = task2(42);
            }
      }
      result = result1 + result2;
}
```

a. Rate your confidence to answer this question with your present knowledge.
   - ☐ A
   - ☐ B
   - ☑ C

22. The following code calculates the dot product for the vectors *A* and *B* with $A, B \in R^n$. Execution of the following code with a single thread produces a correct result. When executing the code with multiple threads, the result is wrong. Explain the issue that occurs in the parallel execution of this code.

```
double s=0;
#pragma omp parallel for
for (i=0; i<n; i++)
{
      s += A[i] * B[i];
}
```

a. Rate your confidence to answer this question with your present knowledge.
   ☑ A
   ☐ B
   ☐ C

23. Write a program that computes the matrix-matrix-multiplication $C = A \cdot B$ where $A, B, C \in R^{N \times N}$ and that is parallelized with OpenMP. State explicitly which variables are *private* or *shared*. Generally aim your solution for best scalability and performance, but do not optimize the caching behavior. Assume $N \geq P$, where P is the number of available processors.

```
void mxm_row(int N, double **C, double **A, double **B) { /* TODO */ }
```

a. Rate your confidence to answer this question with your present knowledge.
   ☐ A
   ☑ B
   ☐ C

24. The code given below illustrates a vector addition: $\vec{b} = \vec{a} + \vec{b}$ with $\vec{a}, \vec{b} \in R^N,\ N \in N$ that also includes a checksum computation. The code shall be executed in parallel with OpenMP on an x86-based cc-NUMA architecture (with Linux or Windows). Comment on the correctness and performance of the code. If the code delivers incorrect results or its performance is improvable, give reasons for the behavior and how to fix it.

```
int i;
const int N; // big value
double checksum = 0.0;
double a[N], b[N];
// initialization
for(i=0; i<N; ++i)
{
      a[i] = i + 2.0;
      b[i] = i * 0.5;
}
// computation
#pragma omp parallel default(none)
for(i=0; i<N; ++i)
{
      b[i] += a[i];
      checksum += b[i];
}
/* .. */
```

a. Rate your confidence to answer this question with your present knowledge.
   ☐ A
   ☐ B
   ☑ C

25. Explain in a few sentences the concept of *thread binding* (also known as *thread pinning*) and why it is needed especially on NUMA architectures.

a. Rate your confidence to answer this question with your present knowledge.
   ☐ A
   ☑ B
   ☐ C

MPI

26. Which two major kinds of communication (in the context of participating processes) do exist in MPI? Explain shortly what the main difference is and give one typical MPI function as example for each communication type (their function parameters are not needed).
    a. Rate your confidence to answer this question with your present knowledge.
       ☑ A
       ☐ B
       ☐ C

27. What does the following pseudo code print that communicates between two processes? Assume input=3.

Process 0	Process 1

`Recv(result)`
`result++;`
`Send(result)`
	`Send(input)`
	`result = 9`
	`Recv(result)`
	`result += 3`
	`Print(result)`

    a. Rate your confidence to answer this question with your present knowledge.
       ☐ A
       ☐ B
       ☑ C

28. What is the output of the following program running with 100 processes?

```
int main(int argc, char**argv) {
      int *sendbuf;
      int recvbuf[100];
      int nprocs, rank;
      MPI_Init(&argc, &argv);
      MPI_Comm_rank(MPI_COMM_WORLD, &rank);
      MPI_Comm_size(MPI_COMM_WORLD, &nprocs);
      sendbuf = (int*) malloc(nprocs*100*sizeof(int));
      for (int i=0; i<nprocs; i++) {
            for (int j=0; j<100;j++)
                  sendbuf[i*100+j] = i;
            }
      }
      MPI_Scatter(sendbuf, 100, MPI_INT, recvbuf, 100, MPI_INT, 0,
                                                MPI_COMM_WORLD);
      if (rank == 3) {
            printf("%d",recvbuf[5]);
      }
}
```

a. Rate your confidence to answer this question with your present knowledge.
   ☐ A
   ☐ B
   ☑ C

29. Explain the parallelization concept of the *Master-Worker* approach. Additionally, state when the application of the master-worker approach makes sense.
    a. Rate your confidence to answer this question with your present knowledge.
       ☑ A
       ☐ B
       ☐ C

30. The following code snippet is part of a SPMD program. For *imax=100*, *jmax=50* and *kmax=20*, how many data elements are used as halo?

```
for (int i=1; i<imax-1; i++) {
      for (j=1; j<jmax-1; j++) {
            for (k=1; k<kmax-1; k++) {
                  A[i][j][k] = A[i-1][j][k] + A[i+1][j][k] +
                               A[i][j-1][k] + A[i][j+1][k] +
                               A[i][j][k-1] + A[i][j][k+1];
            }
      }
}
```

    a. Rate your confidence to answer this question with your present knowledge.
       ☐ A
       ☐ B
       ☑ C

31. The broadcast function distributes information (*data*) from one process (*root*) to all other processes. Describe the algorithm of a broadcast function, which distributes information (*data*) from one process (*root*) directly to all other processes. Consider the MPI specific concept of grouping processes with *communicators*. Use either C code with MPI functions or pseudo code. Commenting your code might help to understand your code.

```
MPI_Bcast(void* data, int count, MPI_Datatype datatype,
          int root, MPI_Comm communicator)
{

}
```

    a. Rate your confidence to answer this question with your present knowledge.
       ☐ A
       ☐ B
       ☑ C

32. Given is the correct MPI program below. This program is executed by exactly two MPI processes. Go through the code and comment on its performance. If the performance can be improved, state why and how it could be adapted to gain the best performance. If you want, you could directly edit the code below.

```
#include <stdio.h>
#include <mpi.h>
#define LEN // defined as a very big value
// definition of functions here
int main(int argc, char **argv)
{
      int rank, nprocs, token, tag = 45;
      int result[LEN];
      MPI_Status status;
      MPI_Request req;
      MPI_Init(&argc, &argv);
      MPI_Comm_rank(MPI_COMM_WORLD, &rank);
      MPI_Comm_size(MPI_COMM_WORLD, &nprocs);
      if (rank == 0) {
            // initialization of result array here (by one process)
            MPI_Ssend (result, LEN, MPI_INT, 1, tag, MPI_COMM_WORLD);
      }
      else { // rank == 1
            MPI_Recv (result, LEN, MPI_INT, 0, tag, MPI_COMM_WORLD,
                                                      &status);
            doLotsOfWorkByRank1();
            doWorkWithResult(result, LEN);
      }
      MPI_Finalize();
      return 0;
}
```

a. Rate your confidence to answer this question with your present knowledge.
   - ☐ A
   - ☐ B
   - ☑ C

Application

33. Explain in a few sentences the goal/ content of the Nanopond application.
    a. Rate your confidence to answer this question with your present knowledge.
       - ☑ A
       - ☐ B
       - ☐ C

34. Describe name, type and size of the main data structures in Nanopond.
    a. Rate your confidence to answer this question with your present knowledge.
       - ☐ A
       - ☑ B
       - ☐ C

35. Describe the characteristics of a Gillespie algorithm.
    a. Rate your confidence to answer this question with your present knowledge.
        ☐ A
        ☑ B
        ☐ C

36. Why would you expect a "struct of arrays" would give better performance than an "array of structs"?
    a. Rate your confidence to answer this question with your present knowledge.
        ☑ A
        ☐ B
        ☐ C

37. Given a Gillespie algorithm operating on a two-dimensional lattice, why is shared-memory linear speedup not possible?
    a. Rate your confidence to answer this question with your present knowledge.
        ☐ A
        ☐ B
        ☑ C

38. Given a Gillespie algorithm operating on a two-dimensional lattice, what prevents linear speedup in the strong-scaling case on distributed-memory machines?
    a. Rate your confidence to answer this question with your present knowledge.
        ☐ A
        ☐ B
        ☑ C

39. Describe how you can find out the upper performance limit (in terms of Amdahl's Law) for the Nanopond application.
    a. Rate your confidence to answer this question with your present knowledge.
        ☐ A
        ☐ B
        ☑ C

40. Describe how to implement an optimizing "compiler" for an esoteric language such as brainf*ck.
    a. Rate your confidence to answer this question with your present knowledge.
        ☐ A
        ☑ B
        ☐ C

Thank you very much for participating!

If you have any further questions, write an e-mail to Barry Rountree (rountree4@llnl.gov) or Sandra Wienke (wienke@itc.rwth-aachen.de).