TENTAMEN / EXAM

TDDC78

Programmering av parallelldatorer / Programming of parallel computers

2015-10-20, 08:00–12:00 TER2

Christoph Kessler
Dept. of Computer and Information Science (IDA)
Linköping University

Hjälpmedel / Allowed aids: Engelsk ordbok / dictionary from English to your native language

Examinator: Christoph Kessler

Jourhavande lärare:
Christoph Kessler (examiner), on travel, 0703-666687
Lu Li (course assistant) 0704-759692; visiting at ca. 10:00 (if possible)

Maximalt antal poäng / Max. #points: 40

Betyg / Grading (prel.): The preliminary threshold for passing (grade 3) is at 20p, for grade 4 at 28p, for grade 5 at 34p.
Because of new regulations by Linköping University, we can no longer give ECTS grades. If you need one, please contact the course secretary after the result has been entered in LADOK.

Tentavisning / Exam review: none for re-exams. Exams will be archived in the IDA student expedition.

General instructions

- Please use a new sheet of paper for each assignment. Order your sheets by assignments, number them, and mark them on top with your exam ID number and the course code.
- You may answer in either English or Swedish. **English is preferred** because not all correcting assistants understand Swedish.
- Write clearly. Unreadable text will be ignored.
- Be precise in your statements. Unprecise formulations may lead to a reduction of points.
- Motivate clearly all statements and reasoning.
- Explain calculations and solution procedures.
- The assignments are *not* ordered according to difficulty.

1. (7.5 p.) **Performance tools and analysis**

   (a) (1 p.) Which performance data collection method is required in order to be able to draw a communication statistics diagram (displaying the amount of bytes communicated between every pair of processes)? Justify your answer (technical reasons).

   (b) (1p) Modern tool-suites for performance analysis of parallel programs consist of a collection of several kinds of tools. Give four different kinds of such tools. *(I.e., no concrete tool names, but a short term saying what each kind of tool does.)*

   (c) (1.5p) Which property/properties of real parallel computers is/are modeled well by the BSP (Bulk-Synchronous Parallel) model, and which property / properties of real parallel computers does it abstract from?

   (d) (1p) When is a parallel algorithm for some problem (asymptotically) *cost-optimal*? *(give a formal definition)*

   (e) (1.5p) What is the difference between relative and absolute parallel speed-up? Which of these is expected to be higher, and why?

   (f) (1.5p) Consider the following (contrived) sequential code:

```
#define N 1000000
float a[N];
...
for (iter=0; iter<1000; iter++) {
  for (i=0; i<N; i++) {
    if (i%2) // even:
       a[i] = a[i] * 1.14;
    else     // odd:
       a[i] = a[i] * 0.89 + 0.5;
 }
}
...
```

      i. Run on a single core of a modern processor, this code takes about 10 seconds. After a simple loop restructuring of the `i` loop, we can perform the same calculation in less than half the time (still single core, single threaded execution). How and why? (1p)

      ii. Can the `i` loop be parallelized? Why or why not? (0.5p)

2. (4 p.) **Parallel program design methodology**

   Foster's design methodology consists of four stages. Name and explain them. Give details about their goals. What are the important properties of the result of each stage? Be thorough!

3. (5 p.) **Parallel computer architecture**

   (a) Given an application with a fixed performance requirement (in GFlops). Why can, provided that the application can be parallelized (work-)efficiently, the transition from a single-core to a multicore execution platform be advantageous from a power efficiency point of view? (1p)

   (b) How does the MSI *write-invalidate protocol* for cache coherence in a bus-based shared memory system work? *(Be thorough)* (2p)

   (c) For a shared memory architecture, does sequential memory consistency imply a deterministic total ordering of all memory accesses? Justify your answer. (1p)

   (d) Give an example of a scalable interconnection network topology (name and sketch) where the node degree is constant, i.e. independent of the number of nodes. What is the advantage of a constant node degree? (1p)

4. (5 p.) **OpenMP**

   (a) Most OpenMP work-sharing constructs, such as the parallel for/do loops, allow to specify an optional `nowait` clause. For example,

```
#pragma omp for nowait
for (i=0; i<N; i++)
  { ... }
#pragma omp for
for (j=0; j<M; j++)
  { ... }
```

   (i) Describe the effect of the `nowait` clause on the computation of the executing processors. (0.5p)

   (ii) Why can it be beneficial for performance to use `nowait`? (0.5p)

   (iii) Formulate a sufficient condition (dependence-based argument) on the two loops in the example above for when it is safe to use `nowait` in the first of the two loops. (1p)

   (b) OpenMP provides different scheduling methods for parallel loops. Characterize the kind of loops that are expected to perform best with *dynamic scheduling*, and explain why. (1.5p)

   (c) What is the memory consistency model guaranteed by OpenMP implementations? (short answer) (0.5p)

   (d) Why is the design of OpenMP helpful for *incremental* parallelization of sequential codes? (1p)

5. (5 p.) **Parallel Basic Linear Algebra**

   (a) (3p) The BLAS (Basic Linear Algebra Subroutines) library API comes in three different levels (BLAS Level 1, 2, 3).

   (i) Name one representative function from each level and give a short description. (1.5p)

   (ii) Why is it very important for a supercomputing center to have efficient implementations of BLAS installed? (0.5p)

   (iii) Which of the levels is most likely to have efficient parallel implementations even where interprocessor communication is expensive, and why? (1p)

   (b) (2p) We discussed two *systolic* parallel algorithms for matrix-matrix multiplication in the lecture.

   (i) What is a systolic parallel algorithm, in general? (1p)

   (ii) How do the systolic algorithms for matrix-matrix multiplication (e.g., Cannon's algorithm) differ from non-systolic algorithms such as SUMMA? (1p)

6. (7.5 p.) **Parallel Solving of Linear Equation Systems**

   Consider the problem of solving a triangular linear equation system, e.g. a system $Ux = b$ where U is a $n \times n$ upper triangular matrix (i.e., all elements below the main diagonal are zero) by *backward substitution*, with the following sequential pseudocode:

   $x_{n-1} \leftarrow b_{n-1}/u_{n-1,n-1};$

   for $i = n-2$ **downto** 0 **do**

   $$x_i \leftarrow \left(b_i - \sum_{j=i+1}^{n-1} u_{i,j} x_j \right) / u_{i,i};$$

   (a) Derive the asymptotic execution time of the above sequential algorithm as an expression in n. (0.5p)

   (b) Suggest a possible parallelization for a distributed memory system with p (single-threaded) nodes, using MPI-like message passing operations.

      i. What kind(s) of parallelism can you exploit here? (0.5p)

      ii. What data distribution would be most suitable for arrays u, x and b in order to maximize parallelism and obtain a reasonable load balancing? Motivate your answer. (1.5p)

      iii. Assume that u and b initially reside on node 0 and the result x should finally be stored completely on node 0 again. Show and explain the resulting (MPI-like) pseudocode, and derive the parallel work, the parallel execution time and the parallel speed-up as expressions in n and p. (You can ignore cache issues for simplicity of analysis, focus on computation and inter-node communication. Use the simple linear delay model $\alpha N + \beta$ for estimating the time of communicating a block of N elements between two nodes.) (5p)

7. (2 p.) **Transformation and Parallelization of Sequential Loops**

Given the following C loop:

```
s = a[0] * b[0];
for (i=1; i<N; i++) {
   s = s + a[i] * b[i];
}
```

(a) Explain why the loop iterations cannot be executed in parallel in this form (dependence-based argument). (0.5p)

(b) What kind of computation does this loop actually do? (0.5p)

Suggest a *parallel algorithm* for the same problem that could utilize up to N processors in parallel (give the algorithm's name, basic idea and asymptotic parallel time complexity, but no details). (1p)

8. (4 p.) **MPI**

(a) (2 p.) Explain the Communicator concept in MPI. How does it support the construction of parallel software components?

(b) (2 p.) *One-sided communication in MPI*

i. Explain the principle of one-sided communication in MPI-2. (1.5p)
Hint: You might want to illustrate your answer with a pseudocode example and/or an annotated picture.

ii. Why is one-sided communication considered being "closer" to the shared-memory programming model than ordinary two-sided message passing? (0.5p)