NTNU – Trondheim
Norwegian University of
Science and Technology

Department of Mathematical Sciences

Examination paper for **TMA4280 Introduction to Supercomputing**

Academic contact during examination: Arne Morten Kvarving

Phone: 97544792

Examination date: Jun 2, 2015

Examination time (from–to): 15:00-19:00

Permitted examination support material: Code: C

All lecture notes, slides, codes, exercises and suggested solutions from course material.
Rottmann: Mathematical formulas.
Earlier exams+suggested solutions in TMA4280.
LINPACK specification and FAQ.
All handwritten notes, including annotations on notes/slides/codes.
Wikipedia printouts.
Introduction to parallel programming from llnl.gov
Simple, approved calculator.

Other information:

- There are 15 questions and 4 points to be earned on each one.
- Make sure to always state any assumptions you make.

Language: English

Number of pages: 6

Number pages enclosed: 0

Checked by:

Date Signature

Problem 1 We have considered two ways of parallelizing a program in the course.

a) Give an overview of the two programming models. In particular, focus on typical issues encountered when using them separately as well as additional things to watch out for when using them in combination.

Fasit: We here expect the student to outline the shared memory and distributed memory approaches. We expect race conditions, deadlocks and the desire to avoid doing MPI calls in threaded sections to be covered.

b) Consider the following code,

```
Vector do_some_funky_stuff(int N)
{
  int i, j;
  Vector result = createVector(N);

#pragma omp parallel for
  for (i=0;i<N;++i) {
    Vector vec = createVector(i);
    for (j=0;j<i;++j)
      vec[j] = func(i,j);
    result[i] = norm2(vec);
    freeVector(vec);
  }

  return result;
}
```

We find that the speedup from using multiple threads is very bad. Explain why. Can you improve it?

Fasit: Speedup is bad due to the varying iteration cost and the static schedule. We should use dynamic or guided.

c) Explain how domain decomposition is implemented within the PETSc framework.

Fasit: See lecture notes and PETSc example codes.

d) Explain the differences between the Fortran and C memory layout. Why do we have to pay attention to this?

Fasit: See notes on BLAS.

Problem 2 We here consider the solution of a 2D Poisson problem with homogenous Dirichlet boundary conditions on a unit square;

$$
\begin{aligned}
-\nabla^2 u &= f \text{ in } \Omega = (0,1)\times(0,1),\\
u\,|_{\partial\Omega} &= 0.
\end{aligned}
$$

The problem is discretized using a second order centered finite difference method, i.e., using the five point formula for the second derivatives on a structured mesh with spacing $h = 1/n$ in both spatial directions. This results in a linear system of equations

$$\mathbf{A}u = f. \tag{1}$$

We solve the problem in parallel using two different algorithms;

1. a FFT based diagonalization approach, and
2. matrix-free conjugate gradient iterations with a block domain decomposition.

The computer is a distributed memory machine interconnected with a network, which we model using the standard linear model,

$$T(b) = \tau_c + \gamma b.$$

Here $T(b)$ is the time to send b bytes over the network, τ_c is a latency and γ is the inverse network bandwith.

a) Give an estimate of the flop count, the memory usage and the parallel speedup for each of the algorithms as a function of the network parameters, the problem size n, the number of processors p and the number of iterations M. You can assume that $mod(n,p) = 0$. Let τ_f denote the time for a single flop.

Fasit:

- Flop count: FST is $n^2\,(2\log n + 2)$. CG is $19n^2 n_{\text{it}}$.
- Memory usage: FST is $2n^2$ doubles (B, B^T), CG is $4n^2$ doubles (4 vectors).

- Speedup:
$$S_P = \frac{T_1}{T_P} = \frac{T_1}{\frac{T_1}{P} + T_{\text{comm}}}.$$
For FFT communication is
$$T_{\text{comm}}^{\text{fft}} = 2\left(\tau_c + \frac{8\gamma n^2}{P}\right).$$
For CG communication is
$$2\tau_c \log_2 P + 4\left(\tau_c + \frac{8\gamma n}{\sqrt{P}}\right).$$
Which means
$$S_P^{\text{fft}} = \frac{\tau_f n^2 (\log n + 1)}{\tau_f \frac{n^2(\log n+1)}{P} + \left(\tau_c + \frac{8\gamma n^2}{P}\right)}.$$
and
$$S_P^{\text{cg}} = \frac{19\tau_f n^2}{\frac{19\tau_f n^2}{P} + 2\tau_c \log_2 P + 4\left(\tau_c + \frac{8\gamma n}{\sqrt{P}}\right)}.$$

b) Let $n = 8192$. You run these two methods using resources in the cloud. You have measured the latency of the network to be approximately $\tau_c = 10^{-4}$s.

You find that the two solvers have the same runtime for $P = 64$. The supplier of the cloud claims that the network has an inverse bandwidth of $\gamma = 10^{-6}$ s/byte. $M = 300$ iterations was required in the CG solver.

Does the data support their claim?

Fasit: Same runtime means that
$$2n^2(\log_2 n + 1)\tau_f + 2\left(\tau_c + \frac{8\gamma n^2}{P}\right) = M\left(\frac{19\tau_f n^2}{P} + 2\tau_c \log_2 P + 4\left(\tau_c + \frac{8\gamma n}{\sqrt{P}}\right)\right).$$
Solving this for γ we find
$$\gamma = \frac{\tau_f\left(\frac{19Mn^2}{P} - 2n^2(\log_2 n + 1)\right) + \tau_c(2M\log_2 p + 4M - 2)}{\left(\frac{16n^2}{P} - \frac{32Mn}{P}\right)}.$$
Plugging in the given data we find $\gamma \approx 6 \times 10^{-7}$. The claim by the provider seems to hold.

c) What distinguishes an SMP from a NUMA machine?

Fasit: Memory access time is uniform on a SMP, non-uniform on a NUMA machine.

Problem 3 For each task in this problem, please choose the correct alternative and give your reasoning. You get 0 points for a wrong answer, 1 point for a correct alternative with the wrong/lacking explanation and 4 points for the correct alternative with a correct explanation.

a) A ccNUMA machine can always do multiple additions in parallel.

Answer: true or false

Fasit: False

A ccNUMA is a cache-coherent non-uniform memory access shared memory machine. It is unrelated to vector operations.

b) In code utilizing dense linear algebra, you have to choose between using BLAS or LAPACK.

Answer: true of false

Fasit: False

BLAS and LAPACK complements each other. LAPACK is built on top of BLAS.

c) MPI-I/O is often used to do post-mortem data assembly.

Answer: true or false

Fasit: False

MPI-I/O is used to avoid post-morten data assembly by writing into a single file.

d) A code with a large parallel efficiency typically has much network traffic.

Answer: true or false

Fasit: False

The use of the network usually reduces parallel efficiency, certainly not the other way around.

e) An LFU cache replacement policy is typically the best for solving partial differential equations.

Answer: true or false

Fasit: False
A LRU cache replacement policy is best due to the data locality in most PDE solvers.

f) A SIMD processor can perform a multiplication and an addition simultanously.

Answer: true or false

Fasit: False
A SIMD processor can perform perform the same operation to multiple datas (SIngle instruction Multiple Data).

g) Cancellation is a concern when subtracting floating point numbers.

Answer: true or false

Fasit: True
If the numbers are close we observe cancellation where very few significant digits are left in the result.

h) Using PETSc there is no point in pre-declaring the sparsity pattern of your operator.

Answer: true or false

Fasit: False
While PETSc will work without specifying the sparsity pattern, building the matrix will be very slow.