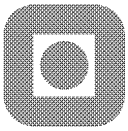
Norwegian University of Science and Technology
NTNU

Department of Computer and Information Science



TDT 4200 Final Exam (Eksamen)
Parallel Computing [Parallelle beregninger/berekningar]
Tuesday, May 26, 2009 [tirdag 26/5-2009]
Time [tid]: 09:00 – 13:00

Instructional contacts during the final[faglige kontakter/faglege kontaktar under eksamen]

Anne C. Elster, 918-97-Thorvald Natvig

ALL ANSWERS NEED TO BE WRITTEN ON THIS EXAM WHERE INDICATED AND THESE SHEETS TURNED IN FOR GRADING. YOU MAY USE THE EXTRA SHEETS PROVIDED FOR THE PROGRAMMING PROBLEM. YOU MAY NOT KEEP OR DISTRIBUTE ANY COPIES OF THIS EXAM.

DISSE EKSAMENSARKENE SKAL INNLEVERES OG ALLE SVAR FØRES INN DER DET ER ANGITT PLASS. KODEOPPGAVEN KAN BENYTTE EKSTRAARKENE VEDLAGT. DET ER IKKE TILLATT Å BEHOLDE ELLER DISTRIBUERE KOPIER AV DETTE EKSAMENSSETTET!

Aids [hjelpemidler]:

Only attached "Summary of MPI Routines and Their Arguments" is permitted as written aid. The note sheet should be turned in with the final exam. No other aids, including calculators, are permitted.

Kun vedlagte "Summary of MPI Routines and Their Arguments" er tillatt som skriftlig hjelpemiddel.. Notatarket skal innleveres med besvarelsen. Ingen andre hjelpemidler, inkludert kalkulatorer, er tillatt.

Grades will be assigned by June 16, 2008. [Karakterer vil bli satt innen 16/6-2008 .]

It is NOT necessary to justify your answer on true/false questions, unless requested.
If there are disagreements between the English and the Norwegian texts, the English text should be used as a guidline.

[Det er ikke nødvendig å avgi forklaring på TRUE/FALSE spørmål der det ikke er bedt om det. Skulle det være uoverstemmelser mellom den engleske og den norske teksten, skal den engelske teksten være førende.]

Written by: _____ **Checked by:** _____

1. WARM-UPS [oppvarming] – TRUE/ FALSE [Sant/Ikke sant--sant/ikkje sant] (10 %)

Circle your answers -- Note: You will get a -1% negative score for each wrong answers and 0 for not answering or circling both TRUE and FALSE.

[Sett sirkel rundt svara -- NB: På denne oppgaven får dere -1% negativt poeng for hvert feilsvar, 0% poeng for å ikke svare eller å sirkle både "TRUE"(sant) og "FALSE" (ikke/ikkje sant).]

a) MPI is ment ot be a platform denpendent interface TRUE/FALSE
(MPI er ment å være et platform-avhenging grensesnitt)

b) Programming in MPI forces you to think about memory issues TRUE/FALSE
(Porgramering i MPI tvinger deg til å tenke på minne)

c) MPI_Send can use wildcards for `source` and `tag` TRUE/FALSE
(MPI_Send kan bruke "åpne variabler" for source og tag)

d) Linear Speedup is usally considered maximum speedup. TRUE/FALSE
(Lineær speedup er vanligvis sett på som maksimum speedup)

e) SIMD instructions are offered on recent Intel Core 2 processors TRUE/FALSE
(SIMD instruksjoner tilbys på moderne Intel Core 2 prosessorer)

f) Data locality matters on NUMA shared memory systems TRUE/FALSE
(Lokasjon av data har noe å si på fellesminnesystemer)

g) Streaming is used to help overcome the memory bottleneck TRUE/FALSE
("Streaming" blir brukt til å overvinne minneflaskehalsen)

h) Blocked MPI calls return when they are locally complete TRUE/FALSE
(Blokkerende MPI kall returnerer når de er ferdige lokalt)

i) Radix sort parallelizes well TRUE/FALSE
(Radixsortering paralleliseres bra)

j) Jacobi iterations typically converge faster than Gauss-Seidel TRUE/FALSE
(Jacobi-iterasjoner kovergerer vanligvis raskere en Gauss-Seidel

2. **MPI BASICS. Fill in the blanks and circle TRUE or FALSE where indicated**.
(fyll inn og sett sirkler rundt true/false hvor indikert) **(10%)**

a. An MPI communciator is always needed TRUE/FALSE
(En MPI communicator er alltid nødvendig)

i. Why/why not? (Hvorfor/hvorfor ikke?)_

_____

_____

b. MPI_ANY_TAG may always be used as an argument**.** **TRUE/FALSE**
(MPI_ANY_TAG kan alltid bli brukt som argument**)**

Why/why not? (Hvorfor/hvorfor ikke?)

_____

_____

c. MPI_COMM_WORLD may always be used as an argument. **TRUE/FALSE**
Why/why not? (Hvorfor/hvorfor ikke?)

_____

_____

d. MPI collective operations do **NOT** use tags **TRUE/FALSE**

Why/Why not? (hvorfor/hvorfor ikke?)

_____

e. It is **illegal** to **alias in/out** arguments in **MPI_Scatter*?*** **TRUE/FALSE**

Why/Why not? (hvorfor/hvorfor ikke?)

_____

_____

_____

3. PARALLEL COMPTING BASICS (10%)

a) Main reasons for super linear speedup is probably:
(Hovedgrunnene til mulig superlinær speedup er sansynligvis :)

_____

_____

b) Amdah´s Law is given by (er gitt som: **) S(p) = p/(1+(p-1)f).**

i) What is S(p) as p → infinity? _____
(Hva blir S(p) hvis p → uendelig/uendeleg**)?**

ii) What is f? (Hva er f?)_____

iii) Mention two ways to overcome Amdah´s Law:
(Nevn to måter/høve å komme seg over Amdahl´s Law på:)

_____

_____

c) Why are Monte Carlo methods easy to parallelize?
(Hvorfor er Monte Carlo metoden lette å parallelisere?)

_____

_____

d) What is the SPMD model? Compare it to SIMD.
(Hva er SPMD modellen? Sammenlign den med SIMD)

_____

_____

4. INTERCONNECTION NETWORKS (10%)

a) Draw a Cluster of shared memory computers with 4 cores on each node
(Tegn en klynge med fellesminne og 4 kjerner per node)

b) Which datastructures can be efficiently mapped to an Ethernet switch?
(Hvilke datastrukturer/ar kan effektivt bli mappet til Ethernet switch?)

c) Which node numbers would be connected to node number 11001 in a 5-D hypercube using Grey code encoiding? (Hvilke nodenumre vil være forbundet med node 11001 i en 5D hyperkube with Greycode numerering)?)

d) Why is wormhole routing and circuit switching preferred over packet swithing? (hvorfor foretrekkes wormhole routing og circuit switching over packet switching?)

e) Why are a hierarchy of swtiches used for very large (>>1000 processor) distributed memory systems? (Hvorfor er et hierarki av switcher brukt i veldig store distributerte systemer?)

5 OPTIMIZATIONS (optimeringer) (10%)

a) **List at least 4 main sources of performance problems with un-optimized codes.**:
[Nevn minst fire grunner til ytelsesproblemer av uoptimerte koder:]

i _____

ii _____

iii

_____

iv

_____

b) Name at least 3 techniques discussed in class for removing branches
(Nevn minst 3 teknikker for å fjerne forgreninger):

_____

_____

_____

c) Which of the following types of branches would one generally optimize:
(Hvilke av de følgende typer forgreninger kan man generelt optimere?)

i) Conditional branches executed for the first time

ii) Conditional branches that have been executed more than once.

iii) Call and Return

iv) Indirect calls & jumps (function pointers & jump tables)

v) Unconditional direct branches/jumps

d) How would you improve the performance of a long switch instruction?
(Hvordan vil du forbedre ytelsen på en lang "switch"instruksjon?)

_____

6. PROGRAMMING AND LIBRARIES (porgramering og bibliotek) **(10%)**

a) **What advantage does intrinsics offer over autovectorization?**
(Hvilke fortrinn har "intrinsics" over auto-vektorisering?)

_____ _____

c) **What are the following three libraries used for?**
(Hva brukes de følgende 3 bibliotekene til?)

Atlas: _____

PLAPACK: _____

PETSc: _____

_____

d) **What is the main diference between MPI_Test and MPI_Probe?**
(Hva er hovedforskjellen/hovudskilnaden på MPI_Test og MPI_Probe?)

_____

_____

e) **What is the main diference between MPI and OpenMP?**
(Hva er hovedforskjellen/hovudskilnaden på MPI og OpenMP?)

_____

_____

f) **What is the main difference OpenMP and POSIX Threads**?
(Hva er hovedforskjellen/hovudskilnaden på OpenMP og POSIX-tråder?)

_____

_____

7. **Performance & Load Balancing (**Ytelse & Lastbalansering**) (10%)**

a) GPUs can be used to accellerate performance of parallel codes. (GPUer kan bli brukt til å aksellerere ytelsen av parallelle koder.

Name two major issues with GPUs that limit their performance and explain what the challenge is: [Nevn to hovergrunner som bergrenser ytelsen til GOUer of forklar hva utfordringene er]

i)_____

ii)_____

b) Name at least 4 static load-balancing techniques
(Nevn minst 4 statiske lastbalanseringsteknikker):

i) _____ii)_____

iii)_____ iv)_____

c) Which conditions need to be satisfied for termination in centralized dynamic load-balancing systems? (Hvilke to kriterier må være oppfylt for at en sentralt dynamisk lastballanert system skal terminere?)

_____

_____

a) List at least three reasons to use library routines whenever possible
[Nevn minst tre grunner til å bruke bibliotek der det er mulig:]

i _____

ii _____

iii _____

8. CUDA PROGRAMMING (programering) (5%)

Insert the missing code in the addElementOnDevice kernel, to do the same as the addElementOnHost function using the following execution configuration:

[Sett in manglene kode i addElementOnDevice kjernen til å gjøre det samme som addElementOnHost funksjonen ved å bruke følgende konfigurasjon:]

```
addElementOnDevice<<<4,25>>>(data_d, 99, 10.0f);

void addElementOnHost(float *data, int N, float element)
{

   int i;

   for(i=0; i<N; i++)
     data[i] = data[i] + element;
}


__global__void addElementOnDevice(float *data, int N, float element)
{

   // YOUR CODE HERE









}
```

9. MPI PROGRAMMING (programering) (15%)

9 a) For Red-Black SoR, each iteration consists of two phases; red and black. After each phase, border data needs to be exchanged. Please fill out the functions with "…" in the following program fragment to do this. Full points will be given for use of persistent, asynchronous communication with only the necessary data transferred, but a working synchronous solution is better than a non-working asynchronous one. Note that you only need to write the functions for the red cells, and you can assume the upper left corner of the local data is red.

```
// 2D Cartesian communicator, already initialized.
MPI_Comm cart;

// Size of local data.
int local_w = local_h = 8;

// With space for border cells, so this is a 10*10 sized array, with the upper left "local" cell at data[10+1]. Already initialized.
float *data;

// Whatever global variables you need
…



// Called after MPI initialization and communicator creation, but before any computation or border exchange.
void init_red_communication() {



}

// Called after red cells are computed, before black cells are computed.
void do_red_communication() {









}

// Called after the problem has converged and we're about to exit.
void free_red_communication() {



}
```

9 b) More MPI programming :

For the program above, make a formula for the total communication time (both red and black) per iteration when using asynchronous communication, using Ts (latency), Beta (inverse bandwidth), w (local height) and h (local height). Show your derivations.

10. OpenMP Programming (10%)

10 a) Parallelize the following function using OpenMP:

```
float sumadd(float *out, float *a, float *b, int n) {
  float s = 0.0f;
  int i;
  for(i=0;i<n;++i) {
    out[i]=a[i] + b[i];
    s += out[i];
  }
  return s;
}
```

10 b) Which conditions would the parameters out, a, b and n have to fulfill in order for the function to be easily optimized with SSE?

EXTRA PAGE (Ekstra-ark)

EXTRA PAGE (Ekstra-ark)