

## **i Information about the exam**

# **Eksamen i INF3380 våren 2017**

**Hjelpemidler: Ett to-sidig A4 ark med håndskrevne notater pluss en kalkulator. Ingen andre hjelpemiddel er tillatt.**

**Alle oppgavene besvares med hjelp av tastatur og mus, det er ikke behov for å bruke digital håndtegning/skisseark.**

Vekting av oppgavene:

Oppgaver 1.1, 1.2 (Work division): 15%

Oppgave 2.1 (One-to-one communication): 10%

Oppgaver 3.1, 3.2, 3.3 (All\_reduce): 20%

Oppgaver 4.1, 4.2 (OpenMP programming): 15%

Oppgaver 5.1, 5.2, 5.3, 5.4 (Dijkstra's algorithm): 40%

## **1.1 Work division; A concrete case**

Det er totalt 20 identiske oppdrag som er uavhengige av hverandre. Det er totalt 6

arbeidere hvor 3 av dem er "vanlige arbeidere" som hver trenger 2 dager til å utføre et oppdrag, mens de andre 3 arbeiderne er "super-arbeidere" som hver bare trenger 1 dag til å utføre et oppdrag. Vi antar at det ikke er mulig å la flere arbeidere jobbe sammen for å utføre et oppdrag raskere.

Hvor mange dager minimum trenger disse 6 arbeiderne til å bli ferdig med alle 20 oppdrag? Begrunn svaret ditt.

**Skriv ditt svar her...**

Format | ↺ | ↻ | ✖

✎ | Σ | ✖

Words: 0

Maks poeng: 10

## 1.2 Work division; Generalisation

Anta nå at det er  $m$  identiske og uavhengige oppdrag, mens det er  $p$  super-arbeidere samt  $p$  vanlige arbeidere. Som før trenger en vanlig arbeider 2 dager til å utføre et oppdrag, mens en super-arbeider trenger bare 1 dag. Vi antar fremdeles at det ikke er mulig å la flere arbeidere jobbe sammen for å utføre et oppdrag raskere.

Forklar hvordan du kan beregne minimum antall dager som trengs av  $p$  vanlige arbeidere pluss  $p$  super-arbeidere til å bli ferdig med  $m$  identiske og uavhengige oppdrag. Utledd en formel.

**Skriv ditt svar her...**

Format ▾ | ↺ | ↻

✎ | Σ | ✕

Words: 0

Maks poeng: 5

## 2.1 Determining the coefficients in the time usage model

Følgende formel

$$t_s + t_w m$$

er ofte brukt til å beregne tidsbruken for å sende en melding av  $m$  bytes fra en prosess til en annen. Hvordan vil du kvantifisere størrelsen til parameterne  $t_s$  og  $t_w$  ved hjelp av eksperimenter og tidsmålinger. (Det er tilstrekkelig å forklare med ord og pseudo koder, ingen konkrete implementasjoner trengs.)

**Skriv ditt svar her...**

Format ▾ | ↺ | ↻

Maks poeng: 10

### 3.1 All\_reduce; A concrete example

Anta at det er totalt 8 MPI prosesser. Hva blir verdien av integer variabel b etter at følgende MPI kodesnutt er kjørt?

```
int my_rank, num_procs, a, b;  
MPI_Comm_rank (MPI_COMM_WORLD, &my_rank);  
MPI_Comm_size (MPI_COMM_WORLD, &num_procs);  
a = num_procs - (my_rank%3);  
MPI_Allreduce( &a, &b, 1, MPI_INT, MPI_MIN, MPI_COMM_WORLD );
```

Begrunn svaret ditt.

**Skriv ditt svar her...**

Format



✎ |  $\Sigma$  | | ✕

---

Words: 0

Maks poeng: 5

### 3.2 All\_reduce; Algorithm

Dersom du skal oppnå samme effekten av å utføre en all\_reduce operasjon, men kun ved hjelp av en serie av en-til-en kommunikasjoner. Forklar, steg for steg, hvordan all\_reduce blir utført ved hjelp av en serie av en-til-en kommunikasjoner på en ring av 8 prosessorer.

**Skriv ditt svar her...**

Format ▾ | | ↺ | | |

✎ |  $\Sigma$  | | ✕

---





Maks poeng: 5

## 4.2 OpenMP assignment 2

Vi har en seriell kodesnutt som følgende:

```
int i, N = 100;
double s = 0.;
double *a = (double*)malloc (N*sizeof(double));
double *b = (double*)malloc (N*sizeof(double));
for (i=0; i<N; i++) {
    a[i] = 10.0+2*i;
    b[i] = 20.0+sin(0.1*i);
}

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

Hva er feilen (eller feilene) i følgende OpenMP versjon? Og hvordan vil du fikse feilen (eller feilene)?

```
int i, N = 100;
double s = 0.;
double *a = (double*)malloc (N*sizeof(double));
double *b = (double*)malloc (N*sizeof(double));

#pragma omp parallel default(shared)
{
    #pragma omp for
    for (i=0; i<N; i++) {
        a[i] = 10.0+2*i;
        b[i] = 20.0+sin(0.1*i);
    }

    #pragma omp for
```





## 5.1 Dijkstras algorithm; Serial implementation

Dijkstras algoritme kan brukes til å finne ut korteste avstand fra en gitt node i en graf til alle andre nodene. "Adjacency" informasjon mellom nodene er antatt å være representert som en distanse-matrise  $w$  som har antall rader og kolonner lik antall noder i grafen. Resultatet av Dijkstras algoritme er en 1D array  $d$  som inneholder korteste avstand fra den angitte kildenoden til alle andre nodene.

Lag en C funksjon som implementerer vedlagte pseudo-kode av Dijkstras algoritme.

**Skriv ditt svar her...**

1	
---	--

Maks poeng: 15

## 5.2 Dijkstras algorithm; A concrete case

Dersom vi har en 6x6 distanse matrise  $w$  med følgende verdier:

```
0 40 15 -1 -1 -1
40 0 20 10 25 6
15 20 0 100 -1 -1
-1 10 100 0 -1 -1
-1 25 -1 -1 0 8
-1 6 -1 -1 8 0
```

Hva blir innholdet av array  $d$ , når vi har kjørt Dijkstras algoritme, med  $s=0$ ? (Det er tilstrekkelig å vise sluttresultatet, ingen mellom regninger trengs.)

**Skriv ditt svar her...**

Format | ↺ | ↻ | ✂ | Σ | ✂

Words: 0

Maks poeng: 10

## 5.3 Dijkstras algorithm; MPI implementation

Lag en MPI implementasjon av Dijkstras algoritme. Til å begynne med antar vi at hele distanse matrisen  $w$  befinner seg kun hos prosess 0. På slutten av MPI implementasjonen

skal neie *a* array'en være tilgjengelig nos prosess u.

**Skriv ditt svar her...**

1

*Syntaks for noen av de viktigste MPI funksjoner:*

```
int MPI_Comm_size( MPI_Comm comm, int *size )
```

```
int MPI_Comm_rank( MPI_Comm comm, int *rank )
```

```
int MPI_Barrier( MPI_Comm comm )
```

```
int MPI_Send(const void *buf, int count, MPI_Datatype datatype,  
             int dest, int tag, MPI_Comm comm)
```

```
int MPI_Recv(void *buf, int count, MPI_Datatype datatype, int source,  
            int tag, MPI_Comm comm, MPI_Status *status)
```

```
int MPI_Bcast( void *buffer, int count, MPI_Datatype datatype, int root,  
             MPI_Comm comm )
```

```
int MPI_Alltoall(const void *sendbuf, int sendcount, MPI_Datatype sendtype,  
                void *recvbuf, int recvcount, MPI_Datatype recvtype,  
                MPI_Comm comm)
```

```
int MPI_Reduce(const void *sendbuf, void *recvbuf, int count,  
              MPI_Datatype datatype,
```



Words: 0

Maks poeng: 5