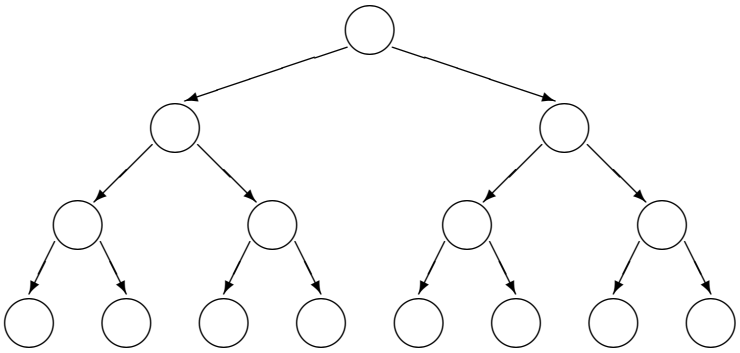
UNIVERSITETET I OSLO

Det matematisk-naturvitenskapelige fakultet

Eksamen i	INF3380 — Parallellprogrammering for naturvitenskapelige problemer
Eksamensdag:	15. juni 2011
Tid for eksamen:	9.00 – 13.00
Oppgavesettet er på 3 sider.	
Vedlegg:	Ingen
Tillatte hjelpemidler:	Kalkulator Ett to-sidig A4 ark med håndskrevne notater

Kontroller at oppgavesettet er komplett før
du begynner å besvare spørsmålene.

Oppgave 1 (vekt 10%)



Figuren over viser 15 arbeidsoppgaver som hver tar 10 minutter å utføre av en arbeider. (En oppgave kan kun utføres av én arbeider.) Hver pil i figuren betyr at oppgaven som blir pekt på ikke kan starte før oppgaven hvor pilen kommer fra er ferdig utført. Hva er korteste tid for 4 arbeidere å gjøre ferdig alle oppgavene? (Du må begrunne svaret ditt.) Gjenta samme spørsmålet når det er 3 arbeidere.

Oppgave 2 (vekt 20%)

Gustafson-Barsis's lov kan uttrykkes som følgende matematiske formel:

$$\Psi(n,p) \leq p + (1-p)s$$

- Hva representerer de matematiske symbolene Ψ, n, p og s?
- Hva kan Gustafson-Barsis's lov brukes til?
- Utledd Gustafson-Barsis's lov.

(Fortsettes på side 2.)

Oppgave 3 (vekt 20%)

```
double trapezoidal (int n) {
  double result = 0.0;
  double h = 1.0/n;
  double x;
  int i;

  x = 0.0;
  for (i=1; i<n; i++) {
    x += h;
    result += exp(5.0*x)+sin(x)-x*x;
  }

  x = 0.;
  result += 0.5*(exp(5.0*x)+sin(x)-x*x);

  x = 1.0;
  result += 0.5*(exp(5.0*x)+sin(x)-x*x);

  return (h*result);
}
```

Lag en parallell versjon av ovenstående `trapezoidal` funksjon ved hjelp av MPI-programmering. (Hint: Den opprinnelige `for`-løkka må modifiseres litt for å få fram parallellitet.) Lag en annen parallell versjon ved hjelp av OpenMP-programmering.

Oppgave 4 (vekt 30%)

Floyds algoritme, som f.eks. kan brukes til å finne korteste avstand mellom to og to byer, benytter følgende trippel `for`-løkke:

```
for (k=0; k<n; k++)
  for (i=0; i<n; i++)
    for (j=0; j<n; j++)
      if ( a[i][j] > (a[i][k]+a[k][j]) )
        a[i][j] = a[i][k]+a[k][j];
```

- Forklar hvor parallellitet befinner seg i `for`-løkken og hvorfor.
- Parallelliser ovenstående `for`-løkke ved hjelp av OpenMP.
- I forbindelse med MPI parallellisering, hvor radene av 2D-array'en `a` er distribuert som "rowwise block stripes", forklar hvor ofte og hvordan MPI kommunikasjonene skjer. (Du trenger ikke å skrive en komplett MPI kode.)

(Fortsettes på side 3.)

Oppgave 5 (vekt 20%)

```
t = 0.0;
while (t < T) {
  t += dt;

  /* compute all the interior points */
  for (i=1; i<=M; i++)
    up[i] = 2*u[i]-um[i]
           +((dt*dt)/(dx*dx))*(u[i-1]-2*u[i]+u[i+1]);

  up[0] = value_of_left_BC(t);   // left boundary point
  up[M+1] = value_of_rigt_BC(t); // right boundary point

  /* preparation for next time step: array shuffle */
  tmp = um;
  um = u;
  u = up;
  up = tmp;
}
```

Ovenstående `while`-løkke implementerer en differansemetode for å løse 1D bølgelikningen (i en lignende stil som oblig 1). Skisser en MPI parallellisering som har mulighet til å gjemme bort kommunikasjonsoverhead ved hjelp av ikke-blokkerende MPI kommandoer for å sende og motta meldinger.