Installasjon av MPI på hjemmeområdet

trulses@student.matnat.uio.no

For de som synes det ubehagelig å vente på Titan så går det an å installere en MPI implementasjon på hjemmeområdet slik at man kan kompilere og kjøre MPI programmer samt få dokumentasjonen til MPI funksjoner gjennom manual kommandoen `man`. Det går hovedsaklig ut på å laste ned og kompilere kildekoden for openmpi, for de som ikke er særlig vant til slikt høres dette nok avskrekkende ut men fortvil ikke, man trenger stort sett bare å copy paste noen enkle kommandoer!

Når man først logger inn på en skole-pc er man i hjemmeområdet sitt, det er vanlig å referere til denne mappen som "∼". Notasjonen vi bruker for en kommando i terminalen er `mappe $ kommando` slik at det blir lite forvirring om hvilke mapper man utfører hvilke kommandoer i, og det er altså det som står etter $ tegnet som skal kjøres i terminalen.

Først av alt er det greit å lage noen rene mapper vi kan arbeide med, vi lager en for å holde kildekoden og en annen for å holde programmene som vi kompilerer, altså `mpicc` og slikt. Så til å begynne kjører vi disse kommandoene i hjemme mappen

```
~ $ mkdir src
~ $ mkdir build
~ $ cd src/
```

Nå befinner vi oss i `src` mappen, vi skal nå laste ned og pakke ut kildekoden for openmpi versjon 1.4.5, dette gjør vi ved kommandoene

```
src/ $ wget http://www.open-mpi.org/software/ompi/v1.4/downloads/openmpi-1.4.5.tar.gz
src/ $ tar -xvzf openmpi-1.4.5.tar.gz
```

Du vil da se masse output, det du ser er navnet på alle filene som blir pakket ut, etter disse kommandoene er kjørt kan du slette `openmpi-1.4.5.tar.gz` men det er greit å holde på den litt til i tilfelle man gjør en feil senere. Nå beveger vi oss inn til kildekoden og setter igang med å konfigurere og kompilere koden, dette gjør vi med kommandoene

```
src/ $ cd openmpi-1.4.5
src/openmpi-1.4.5/ $ ./configure --prefix=$HOME/build
```

Det er veldig viktig at du tar med `--prefix=$HOME/build` når du kjører `configure` kommandoen, du vil nå se masse output og kommandoen vil ta en stund, det som skjer er at `configure` prøver å finne ut hvilken type maskin du har tenkt å kompilere openmpi for. Når `configure` er ferdig med å kjøre kan vi sette igang kompileringen, dette tar også lang tid, du vil kjøre to kommandoer etter hverandre, begge vil ta tid og begge vil gi mye output.

```
src/openmpi-1.4.5/ $ make all -j2
src/openmpi-1.4.5/ $ make install
```

Når disse to er kjørt kan du bevege deg tilbake til hjemmeområdet, alt som står igjen er å sette noe "path variabler" som sier hvor terminalen skal lete etter hvilke filer, for eksempel `mpicc` eller manual siden til `MPI_Scatterv`. Vi setter disse variablene i `~/.bash_login` filen slik at de blir kjørt hver gang vi åpner en ny terminal.

```
src/openmpi-1.4.5/ $ cd
~ $ nano .bash_login
```

På slutten av denne filen må du legge til disse kommandoene

```
export PATH="$HOME/build/bin:$PATH"
export LD_LIBRARY_PATH="$HOME/build/lib"
export MANPATH="$HOME/build/share/man:$MANPATH"
```

Når du er ferdig med å legge til disse linjene trykker du `Ctrl-o` også trykker du enter, etter du har gjort dette trykker du `Ctrl-x` for å gå ut av `nano`, du kan selvfølgelig bruke en annen tekst-editor hvis du er mer fortrolig med det.

Nå skal alt være på plass, men for å være sikker på at ting blir lastet riktig så må du lukke terminalen og så starte en ny. For å sørge for at alt fungerer som det skal er det lurt å kompilere og kjøre et lite eksempel program, for eksempel

```c
#include <stdio.h>
#include <mpi.h>

int main(int argc, char * argv[]) {
  int size;
  int rank;

  int synch = 1;

  MPI_Init(&argc, &argv);
  MPI_Comm_size(MPI_COMM_WORLD, &size);
  MPI_Comm_rank(MPI_COMM_WORLD, &rank);

  MPI_Recv(&synch, 1, MPI_INT,
           rank-1 < 0 ? MPI_PROC_NULL : rank-1,
           101, MPI_COMM_WORLD,
           MPI_STATUS_IGNORE);

  printf("I'm %d out of %d processes!\n",
         rank, size);
  fflush(stdout);

  MPI_Send(&synch, 1, MPI_INT,
           rank+1 >= size ? MPI_PROC_NULL : rank+1,
           101, MPI_COMM_WORLD);

  MPI_Finalize();
  return 0;
}
```

Vi kaller filen for `test.c`, du kompilerer og kjører dette programmet på vanlig måte med disse kommandoene.

```
~ $ mpicc test.c
~ $ mpirun -n 4 a.out
```

Du burde da se noe som dette

```
I'm 0 out of 4 processes!
I'm 1 out of 4 processes!
I'm 2 out of 4 processes!
I'm 3 out of 4 processes!
```

Det kan også være greit å sjekke at manualene er riktig installert, prøv å slå opp `MPI_Recv` med kommandoen

```
~ $ man MPI_Recv
```

Du burde da få opp relevant informasjon om funksjonen `MPI_Recv`, du kan gå ut av manual sider ved å trykke på `q`. Hvis det oppstår problemer så kan du sende meg en e-mail, adressen står på toppen av dette dokumentet.