Universität Hamburg
Department Informatik
Wissenschaftliches Rechnen
Prof. T. Ludwig, M. Kuhn, J. Kunkel

Übungsblatt 4 zur Vorlesung
Hochleistungsrechnen
im SoSe 2010
Abgabe: 10.05.2010

Dieses Übungsblatt soll Ihnen die Möglichkeit geben, Ihre ersten Schritte in der Programmierung mit dem MPI zu machen. Die erworbenen Fertigkeiten werden auf späteren Übungsblätter für komplexere Aufgaben benötigt werden.

1 MPD-Ring und mpiexec (20 Punkte)

In unserem Wiki finden Sie Informationen zum Einbinden des MPI-Moduls und zum Starten eines MPD-Rings.

1. Starten Sie einen MPD-Ring, der die Knoten `node01`, `node02`, `node04`, `node05`, `node07` und `node08` verwendet.
2. Nutzen Sie den Befehl `mpiexec` um den Befehl `hostname` auf jedem der Knoten im MPD-Ring auszuführen.
3. Starten Sie den Befehl aus 2 mehrmals.
   Frage: Was fällt Ihnen auf? Versuchen Sie Ihre Beobachtung zu erklären!

2 Paralleles Starten eines Shell-Scripts (40 Punkte)

1. Erstellen Sie ein Shell-Script `timescript` welches folgendes ausgibt:

   `<HOSTNAME>: <RFC-3339-TIMESTAMP>`

   `<HOSTNAME>`: Einfacher Hostname des Rechners auf dem das Script ausgeführt wird.
   `<RFC-3339-TIMESTAMP>`: Zeitstempel zur Zeit der Ausführung des Scripts im RFC-3339-Format auf die Mikrosekunde genau.

   (Tipp: Sehen Sie sich die Manpages von `hostname` und `date` an.)
2. Führen Sie Ihr Skript mit Hilfe des in der ersten Aufgabe gestarteten MPD-Rings auf allen enthaltenen Knoten gleichzeitig aus.

   Leiten Sie die Ausgabe in eine Datei `timescript.out` um. Sie können dafür eine Pipe (|) in das Kommando `tee` verwenden.

3 Das erste MPI-Programm (150 Punkte)

Erstellen sie ein MPI-Programm `timempi` in C welches eine ähnliche Ausgabe erzeugt wie das parallel gestartete Skript aus der letzten Aufgabe. Dabei sind folgende Vorgaben zu beachten:

- Jeder Prozess mit Rang 1–n soll den String `<HOSTNAME>: <RFC-3339-TIMESTAMP>` bei sich erzeugen und als String per MPI an den Prozess mit Rang 0 senden, welcher die komplette Ausgabe übernimmt.
- Die Ausgabe soll nach Rang der Prozesse geordnet erfolgen.
- Die Prozesse sollen alle erst beenden, wenn die Ausgabe komplett erfolgt ist. Das Programm ist falsch, wenn ein Prozess zu früh beenden könnte!
- Direkt vor dem Beenden soll jeder Prozess einen Text ausgeben: „Rang X beendet jetzt!“ (Tipp: Verwenden Sie `MPI_Barrier(comm)`.)
- Das Programm muss mit beliebig vielen Prozessen lauffähig sein.

4 Ergebnisse sammeln im MPI-Programm (30 Punkte)

Erweitern Sie Ihr MPI-Programm `timempi` zu `timempi2` um folgende Funktion:

- Direkt nach der Ausgabe der empfangenen Strings soll der Prozess mit Rang 0 noch den kleinsten Mikrosekunden-Anteil aller Prozesse ausgeben.

  (Tipp: `MPI_Reduce(...)` bietet sich an.)

5 Beenden des MPD-Rings (5 Punkte)

Damit Ihr MPD-Ring nicht ungenutzt Ressourcen des Cluster verschwendet, beenden Sie ihn wie im Wiki beschrieben.

Abgabe

Als Abgabe erwarten wir ein gemäß den Vorgaben benanntes komprimiertes Archiv (`.tar.gz`), das ein gemäß den Vorgaben benanntes Verzeichnis mit folgendem Inhalt enthält:

- Eine Datei `antwort.txt` mit Ihrer Antwort zu der Frage.
- Das auf dem PVS-Cluster ausführbare Shell-Script `timescript`.
- Die Textdatei `timescript.out` mit der Ausgabe eines Durchlaufs Ihres Scriptes (mit mehreren Prozessen).
- Die Quellen der C-Programme `timempi.c` und `timempi2.c`.
- Ein `Makefile` derart, dass `make timempi`, `make timempi2`, `make clean` und `make` erwartungsgemäße Binärdateien erzeugen bzw. löschen. `make` soll dabei alle Binärdateien auf einmal erzeugen.
- **Keine** Binärdateien.

Senden Sie das Archiv per Mail an: `michael.kuhn@informatik.uni-hamburg.de`.

Rückmeldung (+ 5-10 Punkte)

Bearbeitungszeit			
Schwierigkeit	o zu leicht	o genau richtig	o zu schwer
Lehrreich	o wenig	o etwas	o sehr
Verständlichkeit	o großteils unklar	o teilweise unklar	o verständlich
Kommentar			