

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

Sollten Probleme auftauchen, wenden Sie sich bitte an die Mailingliste der Vorlesung:

`CC-08@pvs.informatik.uni-heidelberg.de`

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`, `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-Skript (40 Punkte)

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

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

`<HOSTNAME>`: einfacher Hostname des Rechners auf dem das Skript ausgeführt wird.

`<RFC-3339-TIMESTAMP>`: Zeitstempel zur Zeit der Ausführung des Skript im RFC-3339-Format auf die Nanosekunde genau.

(Tipp: Manpages `hostname`, `date`)

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!
- Vor dem Beenden soll jeder Prozess direkt 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 Nanosekunden-Anteil aller Prozesse ausgeben.
(Tipp: `MPI_Reduce(...)` bietet sich an.)

5 Beenden des MPD-Ring (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-Skript `timescript`
- die Textdatei `timescript.out` mit der Ausgabe eines Durchlaufs Ihres Skriptes (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 BINÄRDATEI**

Senden Sie das Archiv per Mail an:

`stephan.krempel@pvs.informatik.uni-heidelberg.de`

Rückmeldung (+ 5-10 Punkte)

Bearbeitungszeit			
Schwierigkeit	<input type="checkbox"/> zu leicht	<input type="checkbox"/> genau richtig	<input type="checkbox"/> zu schwer
Lehrreich	<input type="checkbox"/> wenig	<input type="checkbox"/> etwas	<input type="checkbox"/> sehr
Verständlichkeit	<input type="checkbox"/> großteils unklar	<input type="checkbox"/> teilweise unklar	<input type="checkbox"/> verständlich
Kommentar:			