Universität Hamburg, Fachbereich Informatik
Arbeitsbereich Wissenschaftliches Rechnen
Prof. T. Ludwig, Dr. M. Kuhn
M. Blesel, T. Jammer, J. Plehn

Übungsblatt 6 zur Vorlesung
Hochleistungsrechnen
im WiSe 2018/2019
Abgabe: 24.11.2018, 23:59

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

1 Das erste MPI-Programm (150 Punkte)

Erstellen sie ein MPI-Programm `timempi` in C, welches eine ähnliche Ausgabe erzeugt wie das parallele Script von Übungsblatt 3. Dabei sind folgende Vorgaben zu beachten:

- Die Prozesse mit den Rängen 1 bis n sollen den String `HOSTNAME: TIMESTAMP` bei sich erzeugen und als String per MPI an den Prozess mit Rang 0 senden, welcher die komplette Ausgabe übernimmt. (**Tipp**: Sehen Sie sich die Funktionen `gethostname()` und `gettimeofday()` an.)
- 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**: Dazu können Sie `MPI_Barrier()` verwenden.)
- Das Programm muss mit beliebig vielen Prozessen lauffähig sein.

Die Ausgabe könnte wie folgt aussehen:

```
west1: 2012-11-14 13:15:57.968558
west2: 2012-11-14 13:15:57.968557
Rang 1 beendet jetzt!
Rang 0 beendet jetzt!
Rang 2 beendet jetzt!
```

Hinweis: Binden Sie den Header `mpi.h` ein und benutzen Sie den Compiler `mpicc`. Um auf dem Cluster Zugriff auf den MPI-Compiler zu erhalten, müssen Sie das entsprechende Paket zuerst mit `spack load mvapich2` laden.

2 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**: Hierfür können Sie `MPI_Reduce()` verwenden.)

Die Ausgabe könnte dann wie folgt aussehen:

```
west1: 2012-11-14 13:15:57.968558
west2: 2012-11-14 13:15:57.968557
968557
Rang 2 beendet jetzt!
Rang 0 beendet jetzt!
Rang 1 beendet jetzt!
```

3 Paralleles Debugging mit DDT (60 Punkte)

In dieser Aufgabe sollen Sie sich mit dem parallelen Debugger DDT vertraut machen. Verwenden Sie dazu Ihr gerade geschriebenes paralleles Programm.

Unter folgendem Link finden Sie eine kleine Einführung in DDT:

`https://wr.informatik.uni-hamburg.de/teaching/ressourcen/debugging#ddt`

Hinweis: Für die Benutzung von DDT benötigen Sie X-Forwarding. Alternativ können Sie X2Go nutzen, welches üblicherweise eine höhere Leistung erreicht.

Die Lizenz auf dem Cluster ist für 16 gleichzeitige Prozesse gültig. Bedenken Sie bei Ihren Tests, dass sich alle Benutzer diese Anzahl teilen! Das Aufrufen von DDT ist nur vom Master-Knoten möglich, wobei ein Programm durchaus auf mehreren Knoten analysiert werden kann.

Machen Sie sich ein wenig mit DDT vertraut, indem Sie Ihr Programm auf zwei Knoten mit jeweils zwei Prozessen debuggen. Dokumentieren Sie folgende Punkte mit jeweils einer kurzen Beschreibung und einem Screenshot; markieren Sie die relevanten Stellen der Screenshots:

- Wie kann man in DDT die Programmparameter angeben? Gibt es dafür auch andere Möglichkeiten? Wenn ja, welche?
- Setzen Sie in einer Zeile einen Breakpoint. Welche Step-Möglichkeiten gibt es und wie unterscheiden sich diese?
- Schauen Sie sich die Werte der Variable an, in der Sie den Rang des aktuellen Prozesses gespeichert haben. Was fällt Ihnen in der Darstellung auf? Vergleichen Sie die Werte aller Prozesse mit Hilfe des Rechtsklickmenüs.
- Machen Sie sich mit der Funktion des Evaluate-Fensters in der rechten unteren Ecke vertraut.
- In der oberen Leiste finden Sie eine Übersicht aller Prozesse und Threads Ihres Programmes. Wechseln Sie zwischen den einzelnen Prozessen und beobachten Sie das Evaluate-Fenster.
- Erweitern Sie Ihr Programm um ein Array und initialisieren Sie es mit beliebigen Zahlenwerten. Lassen Sie sich die Werte anzeigen. Welche sonstigen Visualisierungsmöglichkeiten bietet DDT? (**Hinweis:** Das Array wird nur für diese Aufgabe benötigt und soll in den abgegebenen Programmen `timempi` bzw. `timempi2` **nicht** enthalten sein.)

Abgabe

Als Abgabe erwarten wir ein gemäß den Vorgaben benanntes komprimiertes Archiv, das ein gemäß den Vorgaben benanntes Verzeichnis mit folgendem Inhalt enthält:

- 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!
- Ein PDF `ddt.pdf` mit den Antworten und Screenshots.

Senden Sie das Archiv an `hr-abgabe@wr.informatik.uni-hamburg.de`.