Praktikum 3: Intro: OpenMP

M.Thaler, 2/2016, ZHAW

1 Einführung

In diesem Praktikum werden sie die grundlegenden Direktiven zur Parallelisierung von Programmen mit OpenMP kennenlernen. Das GOMP Projekt hat für die GNU Compiler Collection eine OpenMP Implementation entwickelt, die wir hier verwenden werden. Dazu muss bei den Compiler Flags `-fopenmp` angegeben werden und die gomp-Biblothek mit `-lgomp` dazugelinkt werden.

2 First Steps

Im Verzeichnis `introMP/a1` finden sie (wohl zum zigsten Mal) ein Hello World Programm. Anhand dieses einfachen Programms möchten wir sie mit den Grundkonzepten vertraut machen, wie in OpenMP Programm-Code parallelisiert wird. Notieren sie sich jeweils zu einzelnen Punkten die Resultate und was Ihnen auffällt.

1. Übersetzen sie das Programm mit `make` und starten sie `main.e` (... das Resultat haben sie wohl erwartet).

2. Parallelisieren sie nun das Programm wie folgt mit einer *Parallelen Region*:

   ```
   #pragma omp parallel
   {
       printf("Hello World from Thread\n");
   }
   ```

   Starten sie `main.e` und analysieren sie die Ausgabe.

3. Die Anzahl Threads kann mit Bibliotheksfunktionen gesetzt und resp. gelesen werden:

   - `int omp_get_num_threads()`
   - `omp_set_num_threads(int p)`

   Lesen sie im Hauptprogramm **vor**, **in** und **nach** der parallelen Region die Anzahl Threads und geben sie sie auf jeweils auf dem Bildschirm aus.

4. Setzen sie nun auf der ersten Zeile im Hauptprogramm die Anzahl Threads auf 5: analaysieren sie die Ausgabe.

5. Die Anzahl Threads kann auch mit einer `clause` in der parallel Direktive `parallel` gesetzt werden (löschen sie aber den Funktionsaufruf `omp_set_num_threads(5)` nicht):

   ```
   #pragma omp parallel num_threads(7)
   {
       ...
   }
   ```

   Was gibt das Programm nun aus und was lässt sich daraus schliessen?

6. Wechseln sie nun ins Verzeichnis `introMP/a2`. Hier finden sie das Programm `main.c`, das "Hello World" in einer for-Schleife ausgibt und mehrere auskommentierte Zeilen enthält.

   Führen sie unten stehende Versuche mit aktivierten und auskommentierten Programmzeilen durch und dokumentieren sie die Resultate. Beachten sie speziell wie die Iterationen der for-Schleife auf die Threads abgebildet werden.

Versuch	aktivierte Zeile	deaktivierte Zeilen	Anzahl Iterationen
a)	keine	1, 2, 3	4
b)	1	2, 3	4
c)	2	1, 3	4
d)	3	1, 2	4
e)	3	1, 2	8
f)	1, 2	3	4
g)	1, 3	2	4

2.1 Fazit

OpenMP ist kein automatisches paralleles Programmiermodell: Parallelismus ist explizit und liegt in der Verantwortung des Programmierers.

3 Parallelisierung mit OpenMP

Im Verzeichnis `introMP/a3` finden sie eine serielle Implementation der Berechnung von π inklusive Zeitmessung mit der OpenMP-Funkion `omp_get_wtime()`.

1. Parallelisieren sie die serielle Implementation der Berechnung von π mit Hilfe von OpenMP. Bestimmen sie Speedup und Effizienz für $1...N$ Threads, N = Anzahl CPUS.

2. Verwenden sie nun einen SIMD-Loop und bestimen sie erneut Speedup und Effizienz. Vergleichen sie die Resultate mit denen aus obiger Aufgabe.

4 Take Home

Was ist bei der Parallelisierung von seriellen Programmen mit OpenMP alles zu beachten?

Notieren sie sich die wichtigsten Punkte, die sie im Zusammenhang mit diesem Praktikum kennengelernt haben.