Universität Hamburg
Fachbereich Informatik
Arbeitsbereich Wissenschaftliches Rechnen
Michael Kuhn, Kira Duwe

Übungsblatt 7 zur Vorlesung
Hochleistungs-Ein-/Ausgabe
im SoSe 2017
Abgabe: 05.07.2017, 08:00

1 Nicht-zusammenhängende Ein-/Ausgabe (180 Punkte)

In den Materialien finden Sie ein Benchmark-Programm, das mit Hilfe mehrerer MPI-Prozesse parallel Daten über die POSIX- und MPI-IO-Schnittstellen schreibt und liest. Das Programm ist dabei in Schreib- und Lese-Phasen aufgeteilt, die vollständig voneinander getrennt sind.

Die Phasen sind dabei wiederum in Iterationen unterteilt. In jeder Iterationen findet genau ein E/A-Zugriff statt, wobei immer zehn Iterationen eine logische Einheit bilden.

Das Benchmark-Programm arbeitet mit einer gemeinsamen Datei, auf die alle Prozesse gleichzeitig zugreifen. Das Zugriffsmuster sieht dabei wie folgt aus:

Prozess	0	1	...	n	...	0	1	...	n
Iteration	0	0	...	0	...	m	m	...	m

Modifizieren Sie das Benchmark-Programm so, dass die E/A-Zugriffe von jeweils zehn Iterationen mit Hilfe einer einzigen nicht-zusammenhängenden E/A-Operation durchgeführt werden. Für POSIX können Sie die `aio_write`- und `aio_read`-Funktionen nutzen, für MPI-IO müssen Sie eine entsprechende Dateisicht konstruieren. Stellen Sie sicher, dass keine nicht benötigten Daten gelesen bzw. geschrieben werden.

Implementieren Sie für MPI-IO außerdem die Möglichkeit kollektive E/A zu benutzen.

2 Leistungsmessung (120 Punkte)

Vergleichen Sie die Leistungsfähigkeit individueller zusammenhängender E/A-Zugriffe, individueller nicht-zusammenhängender Zugriffe und kollektiver nicht-zusammenhängender E/A, wobei Letzteres nur mit MPI-IO möglich ist. Nutzen Sie dafür das auf dem Cluster verfügbare Lustre-Dateisystem, das unter `/mnt/lustre` gemountet ist.

Rufen Sie das Benchmark-Programm dabei wie folgt mit acht Prozessen pro Knoten auf, wobei für MPI-IO entsprechend `--mpi` benutzt werden muss:

```
$ mkdir /mnt/lustre/${USER}/07
$ mpiexec ./benchmark --posix --path=/mnt/lustre/${USER}/07   \
                      --sync-on-close --repetitions=3         \
                      --block-count=$count --block-size=$size \
                      --command 'sudo /home/hr/drop-caches.sh'
```

Hinweis: Um das Benchmark-Programm kompilieren zu können, müssen Sie dessen Abhängigkeiten mit `spack load -r glib mpi` laden.

Bestimmen Sie auf Basis des oben angegebenen Zugriffsmusters die optimale Blockgröße und wählen Sie die Blockzahl so, dass jeder Prozess 128 MiB Daten schreibt und liest. Stellen Sie dabei sicher, dass die Zugriffe des Benchmarks für optimale Leistung an den Streifen ausgerichtet sind. Die Streifenbreite beträgt standardmäßig 1 MiB und kann mit `lfs` geändert werden.

Sammeln Sie Leistungsergebnisse für 1–10 Knoten und visualisieren Sie diese.

Abgabe

Erstellen Sie ein Verzeichnis mit ihrem C-Programm `benchmark.c`, dem dazugehörigen Makefile und der Datei `auswertung.pdf`. Packen Sie ein komprimiertes Archiv aus dem sauberen Verzeichnis (ohne Binärdateien).

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