HLR - Hochleistungsrechnen

Aufgabenblatt 11

Merle Hoffmann, Joël Miramon, Max Press

Für die Bearbeitung dieses Aufgabenblatts wurde nicht unser eigenes paralleles Programm aus der vorherigen Abgabe verwendet. Wir haben freundlicherweise das Programm von Herrn Ott - aus Kjell's Gruppe - zur Verfügung gestellt bekommen, welches die Bedingungen für das letzte Blatt zu erfüllen schien. Leider mussten wir nach langen warten feststellen, dass dieses Programm auch nicht fehlerfrei war, welches wir in den betroffenen Stellen vermerkt haben. **1.1: Weak Scaling Gauss-Seidel:**

Anschließend sind die Ergebnisse für Gauss-Seidel, welches bei uns leider nur auf den ersten beiden Knoten funktioniert hat. Was bei den limitierten Ergebnissen zu sehen ist, dass wenn man die nProcs pro Knoten erhöht, so wird die Berechnungszeit etwas kleiner.

nProcs	nNodes	Interlines	Zeit	nProcs/nNodes
1	1	400	285.1031 s	1
2	1	564	257.4608 s	2
4	2	800	N/a s	2
8	4	1128	N/a s	2
16	4	1600	N/a s	4
24	4	1960	N/a s	6
64	8	3200	N/a s	8

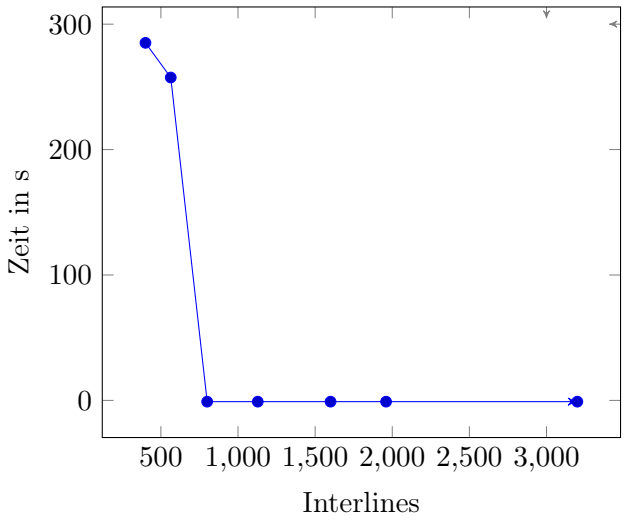

Abbildung 1: Weak Scaling Gauss-Seidel

In Abbildung 1 sieht man die Ergebnisse aus Tabelle 1, mit den Interlines auf der x-Achse und der Zeit in s auf der y-Achse. Es ist anzumerken, dass bei 564 Interlines, die nProcs/nNodes den Faktor 2 hat, was wohl wahrscheinlich verantwortlich für die schnellere Berechnungszeit, im Vergleich mit dem Faktor 1 bei 400 Interlines.

1.1: Weak Scaling Jakobi:

Anschließend sind die Ergebnisse für Jakobi, welche bei uns zwar auch auf mehreren Knoten funktioniert hat, aber leider nur in einer Ausführung, deswegen konnte keine Standardabweichung mit eingezeichnet werden.

nProcs	nNodes	Interlines	Zeit	nProcs/nNodes
1	1	400	288.4214 s	1
2	1	564	277.7606 s	2
4	2	800	561.1149 s	2
8	4	1128	1116.5222 s	2
16	4	1600	1135.0017 s	4
24	4	1960	1130.6812 s	6
64	8	3200	2301.0263 s	8

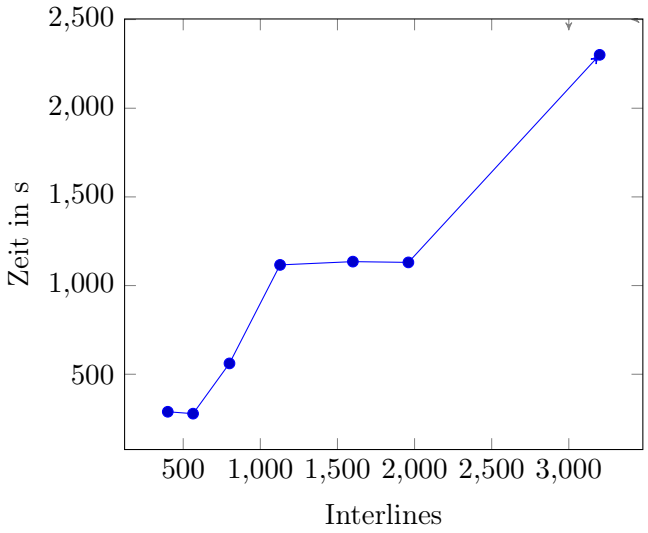

Abbildung 2: Weak Scaling Jakobi

Was man bei den Ergebnissen sehen kann, ist dass die Berechnungszeit mit zunehmendem nNodes zunimmt, vermutlich, weil die Kommunikation zwischen den Nodes langsam ist. Außerdem ist die Berechnungszeit bei gleichbleibendem nNodes nahezu konstant, weil die Interlines und nProcs so gewählt sind, dass jeder Prozess immer gleich viel rechnen muss.

Wie erklären Sie sich die Wahl der Interlines in Bezug auf die Prozesszahl?:

Die Interlines wurden so gewählt, dass $\frac{\texttt{Interlines}^2}{\texttt{nProcs}}$ immer ungefähr ein konstanter Wert bleibt, in diesem Fall ca. 160000. Das sorgt dafür, dass jeder Prozess ungefähr immer die gleiche Anzahl an Zeilen zu berechnen hat.

1.2: Strong Scaling Gauss-Seidel:

Anschließend sind die Ergebnisse für Gauss-Seidel, welche bei uns leider nur auf den ersten beiden Knoten funktioniert hat. Was bei den limitierten Ergebnissen zu sehen ist, dass bei gleichbleibenden Interlines, aber verdoppelten nProcs und nNodes, die Berechnungzeit fast halbiert wird, obwohl die Anzahl von Prozessen pro Knoten gleich bleibt.

nProcs	nNodes	Interlines	Zeit	nProcs/nNodes
12	1	1920	503.0360 s	12
24	2	1920	254.5545 s	12
48	4	1920	N/a s	12
96	8	1920	N/a s	12
120	10	1920	N/a s	12
240	10	1920	N/a s	24

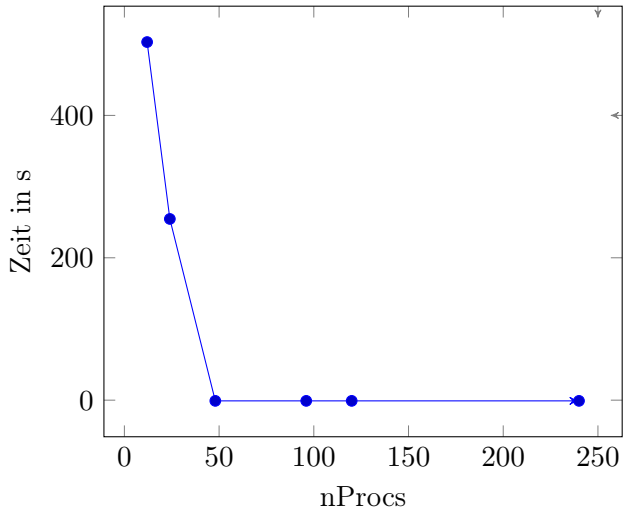

Abbildung 3: Strong Scaling Gauss-Seidel

1.2: Strong Scaling Jakobi:

Anschließend sind die Ergebnisse für Jakobi, welche auch wieder nur einmal berechnet wurden, und deswegen keine Angabe für die Standardabweichung gemacht werden kann. Was aber auf ersten Blick zu sehen ist, dass im Gegensatz zu Gauss-Seidel's Strong Scaling, keine Halbierung der Zeit bei Verdoppelung der Prozessen und Knoten zu beobachten scheint, nur ein „Speedup" von etwa einer Sekunde, was durchaus unbedeutend erscheint. Mit weiterer Verdopplung von nProcs und nNodes, aber Beibehaltung von Interlines, nimmt die Zeit zu bis zu dem Zeitpunkt wo das Verhältnis von nProcs zu nNodes von 12 auf 24 verdoppelt wird, wo man dann ein Speedup von etwa 20 Sekunden sehen kann. Im Vergleich zu den ersten zwei Werten von Gauss-Seidel, ist dieser Speedup bedeutend geringer, obwohl die Anzahl nProcs deutschlich größer ist.

nProcs	nNodes	Interlines	Zeit	nProcs/nNodes
12	1	1920	275.9095 s	12
24	2	1920	274.7334 s	12
48	4	1920	277.0955 s	12
96	8	1920	284.6428 s	12
120	10	1920	288.7633 s	12
240	10	1920	254.3021 s	24

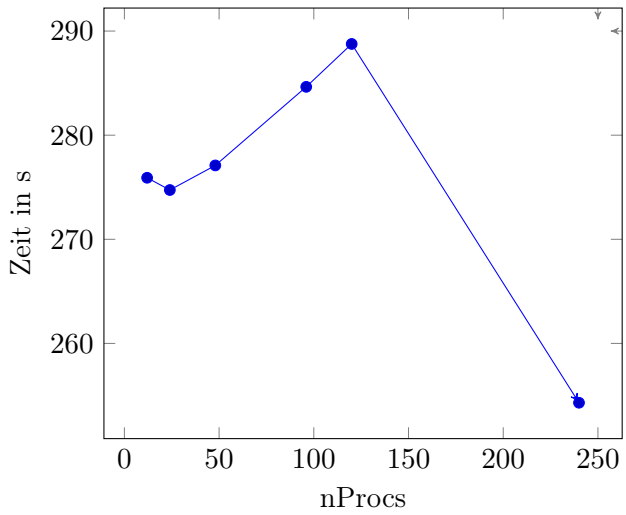

Abbildung 4: Strong Scaling Jakobi

1.3: Kommunikation Gauss-Seidel:

Anschließend sind die Ergebnisse für Gauss-Seidel, welche bei uns leider nur auf einen Knoten funktioniert haben. Wir haben einen Wert bekommen. Daraus lässt sich nicht viel interpretieren.

nProcs	nNodes	Interlines	Zeit	nProcs/nNodes
10	1	200	124.1704 s	10
10	2	200	N/a s	5
10	3	200	N/a s	3.3333
10	4	200	N/a s	2.5
10	6	200	N/a s	1.6666
10	8	200	N/a s	1.25
10	10	200	N/a s	1

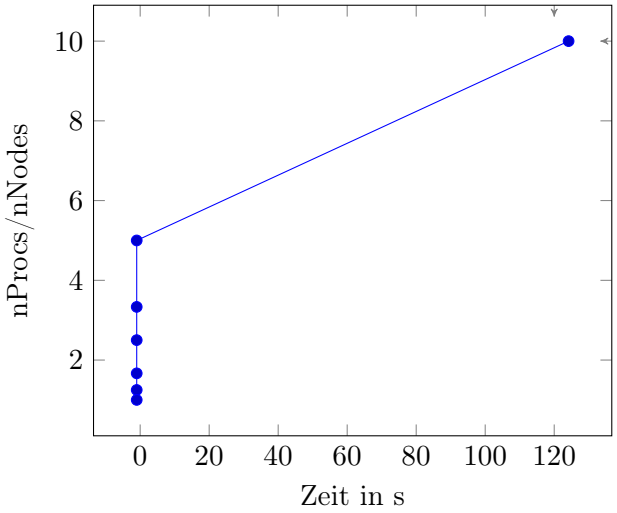

Abbildung 5: Kommunikation Gauss-Seidel

1.3: Kommunikation Jakobi:

nProcs	nNodes	Interlines	Zeit	nProcs/nNodes
10	1	200	124.1704 s	10
10	2	200	261.3004 s	5
10	3	200	401.5337 s	3.3333
10	4	200	530.1211 s	2.5
10	6	200	778.2033 s	1.6666
10	8	200	776.2203 s	1.25
10	10	200	1350.9431 s	1

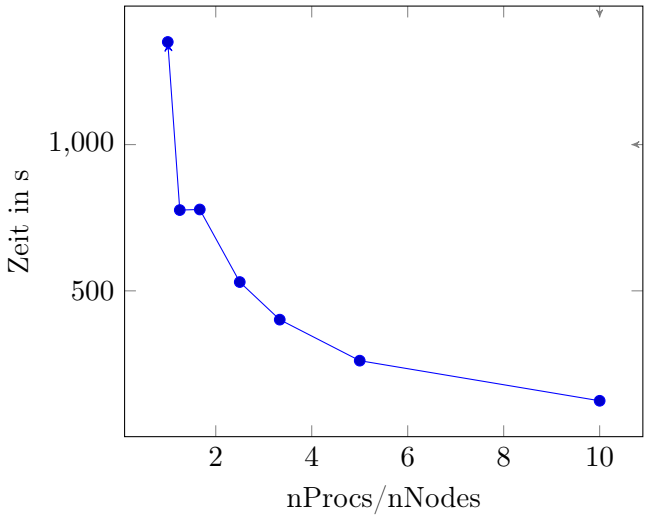

Abbildung 6: Kommunikation Jakobi

Bei diesen Ergebnissen ist leicht zu erkennen, wie die Kommunikation zwischen den Nodes die Berechnungszeit erhöht. Die Berechnungszeit nimmt mit zunehmendem nProcs/nNodes immer weiter ab.