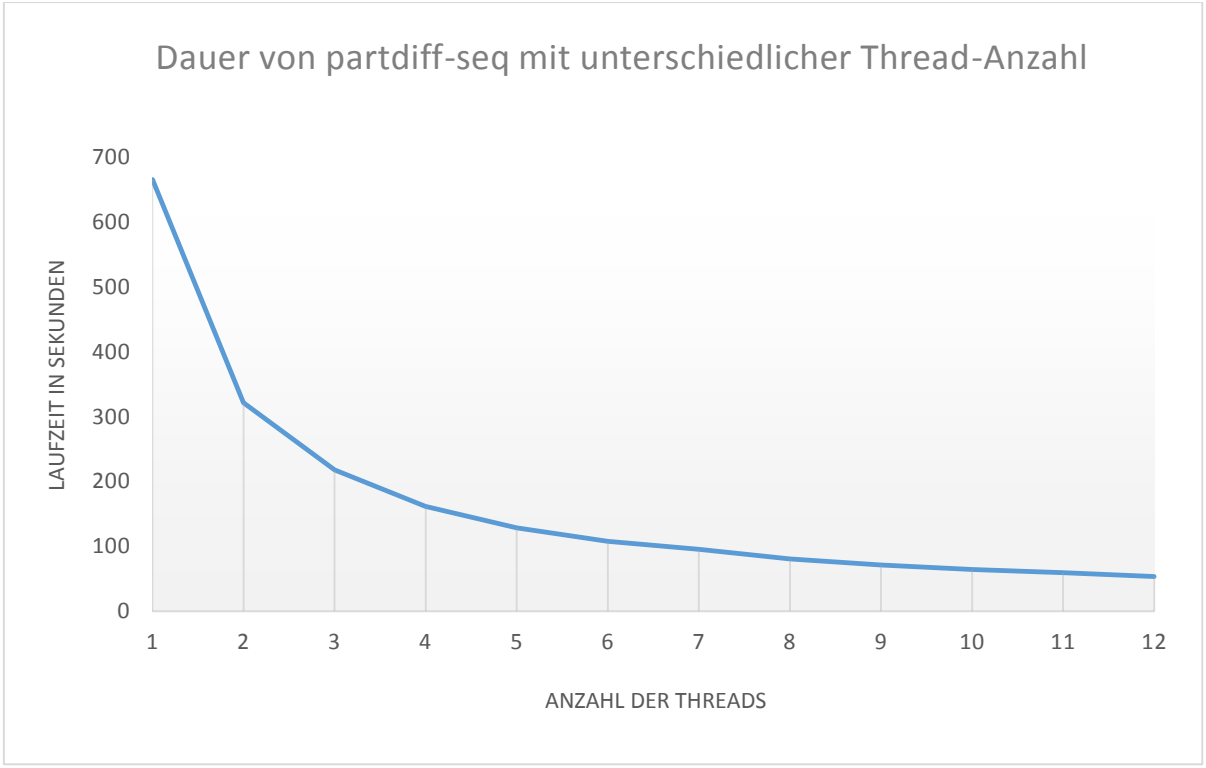
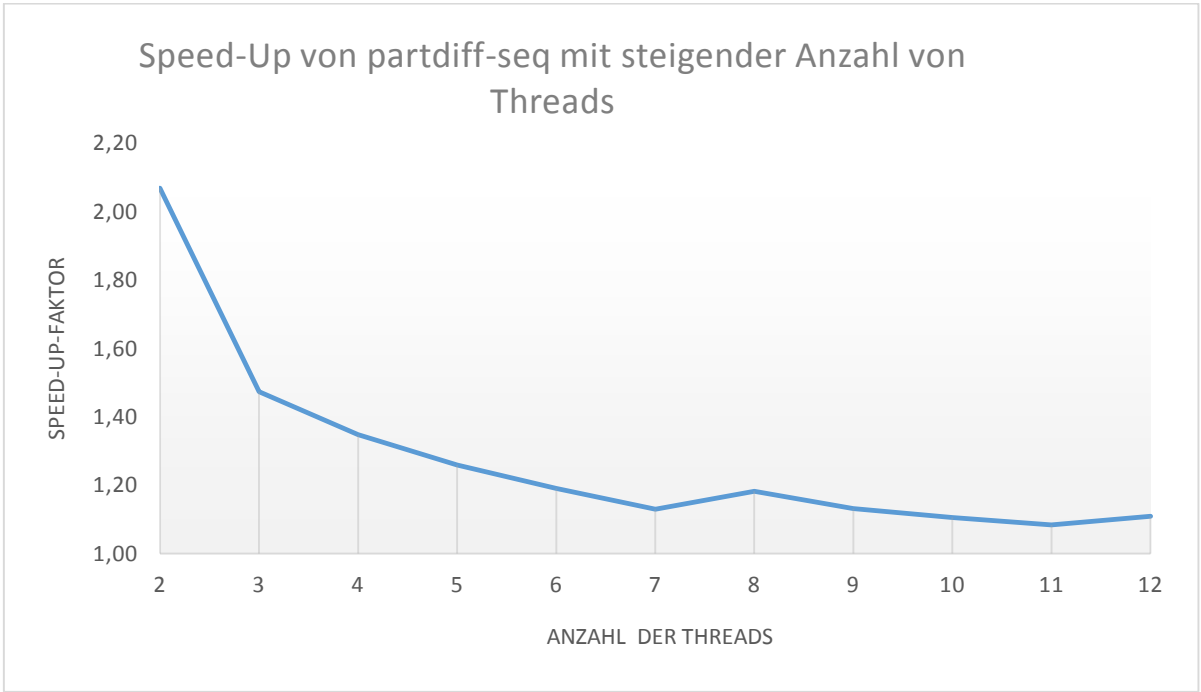
14.11.2014
Hochleistungsrechnen
Timon Back
Fabian Behrendt

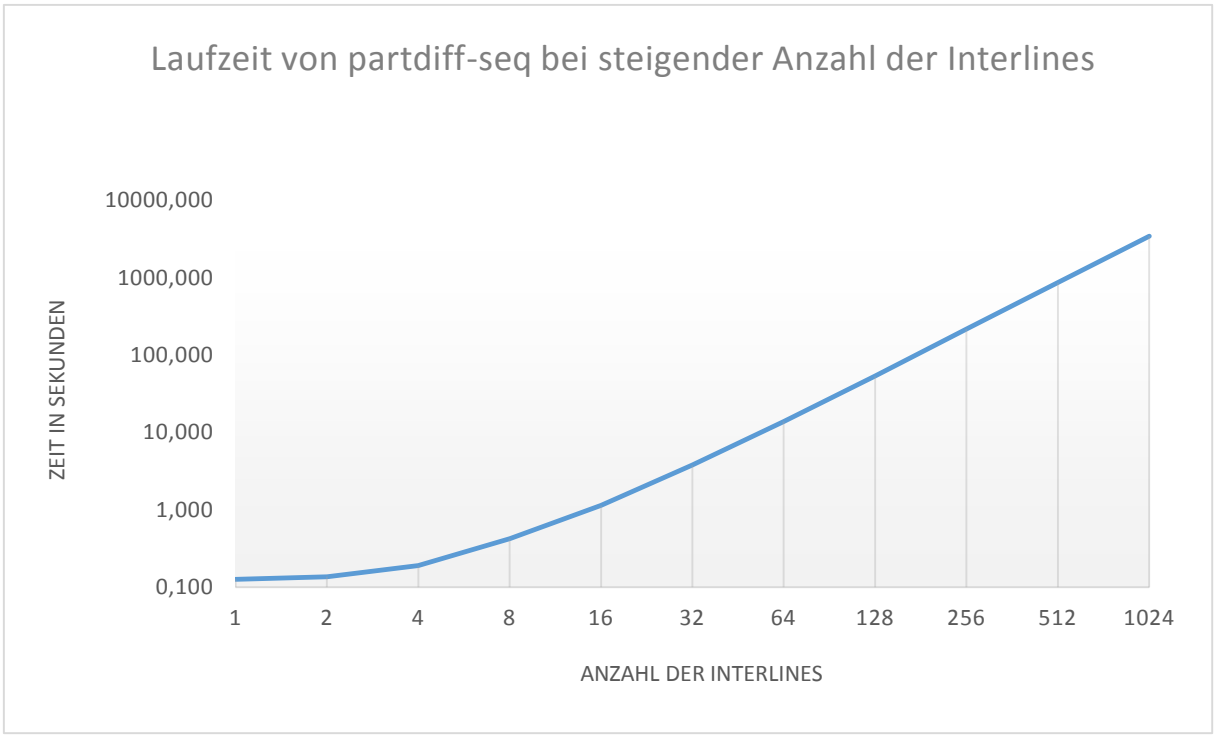
Diagramme zu Messung 1:

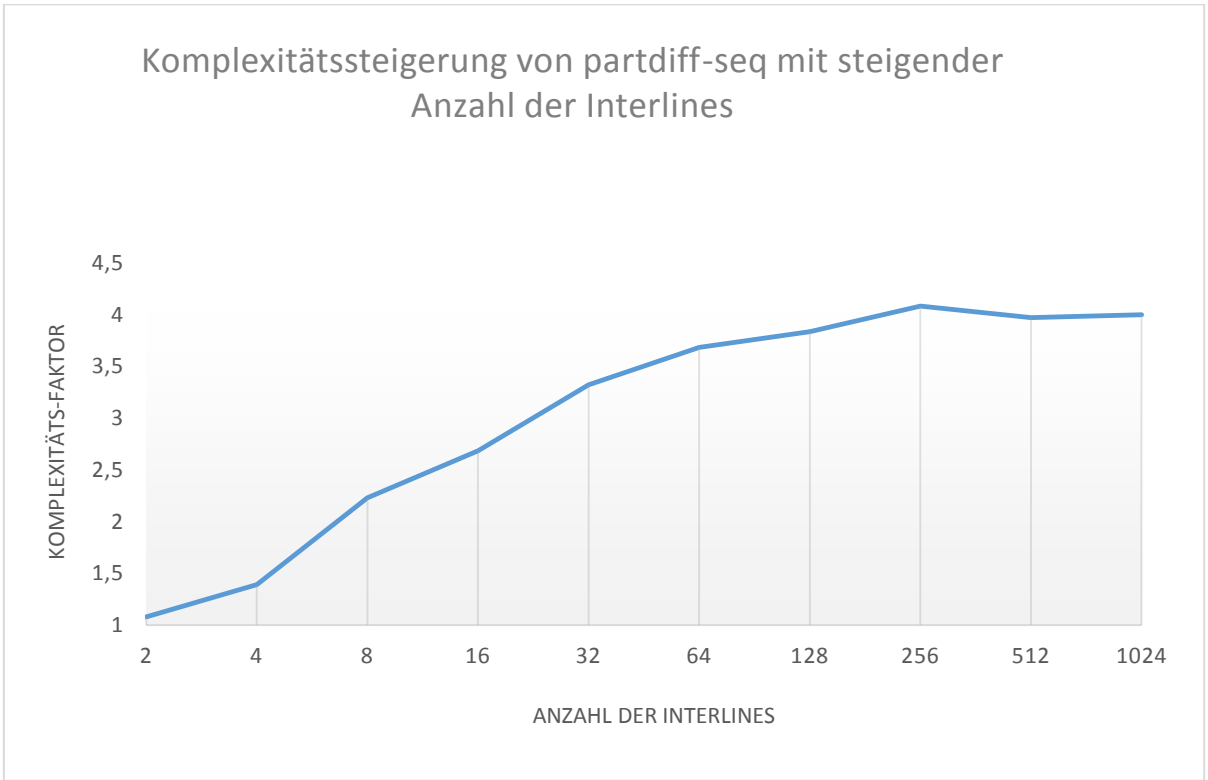




14.11.2014
Hochleistungsrechnen
Timon Back
Fabian Behrendt

Diagramme zu Messung 2:





14.11.2014
Hochleistungsrechnen
Timon Back
Fabian Behrendt

Interpretation der Ergebnisse

Messung 1:

In Messung 1 wurde die Laufzeit von partdiff-seq mit Verwendung von ein bis zwölf Threads mit 512 Interlines gemessen. Es lässt sich feststellen, dass sich bei Verdopplung der Threads die Programmlaufzeit halbiert. Somit lässt sich sagen, dass dieses Programm (begrenzt) auf mehrere/viele Threads skalieren lässt. Große Geschwindigkeitsgewinne bringen die ersten zusätzlichen Threads, während die Verwendung eines weiteren Threads bei bereits 10 oder 11 Threads nur noch einen Beschleunigung des Programms im niedrigen zweistelligen Prozentbereich bewirkt.

Messung 2:

In Messung 2 wurde partdiff-seq mit 12 Threads ausgeführt unter Variation der Interlines (1-1024). Mit steigender Anzahl der Interlines steigt auch die Programmlaufzeit an. Zu Beginn weniger stark, jedoch mit einer Interline-Zahl von größer als 128 annährend konstant mit dem Faktor 4. Sprich: Doppelte Interlines-Zahl, vierfache Programmlaufzeit.

Unsere Vermutung ist, dass bei einer kleinen Anzahl an Interlines, die Berechnung nicht so viel Zeit ausmacht, wie der Rest des Programms, z.B. die Reservierung und Initialisierung von Speicher.

Wenn die Vorbereitung und Ausgabe der Berechnung keine Zeit kosten würde, dann würde die Laufzeit konstant um den Faktor 4 steigen. Im Diagramm stellt sich das bei der Komplexitätssteigerung durch eine Gerade mit f(x)=4 und die Laufzeit (auf einer logarithmischen Skala) mit einer perfekten Parabel dar.