# Hochleistungsrechnen - Übungsblatt 11

Joscha Fregin, Tim Kilian, Stefan Knispel

25. Januar 2018

# 1 Verwendung von 3 Prozessen und 2 Knoten

# 1.1 Jacobi-Verfahren

# 1.1.1 Startphase des Programms

Hier ist schön zu sehen, dass der Prozess 1 nach der Initialisierung seine Berechnung startet. Der Prozess 2 wartet nach seiner Initialisierung auf den Prozess 1 und fängt nach dem Empfangen auch an zu rechnen. Der dritte Prozess wartet auf den zweiten und rechnet nach Empfang ebenfalls los.

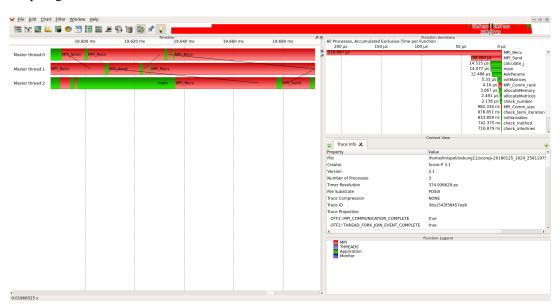


Abbildung 1: Jacobi-Verfahren mit 3 Prozessoren und 2 Knoten : Startphase

# 1.1.2 Phase der Synchronisation

Hier ist schön zu sehen, dass die einzelnen Prozesse ihre Berechnung kurz unterbrechen, um die erste schon berechnete Zeile an den darüberliegenden Prozess zu schicken. Danach rechnen sie weiter und schicken am Ende ihre letzte Zeile an den unteren Prozess.

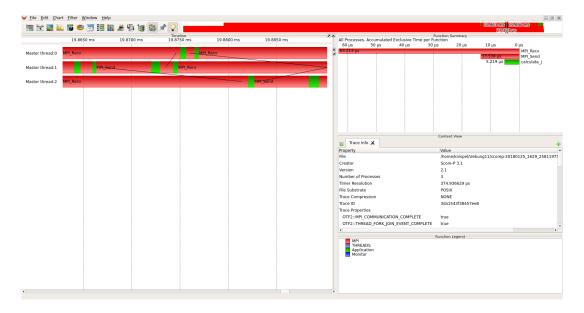


Abbildung 2: Jacobi-Verfahren mit 3 Prozessoren und 2 Knoten : Phase der Synchronisation

#### 1.1.3 Phase des Einsammelns

Bei allen drei Prozessen ist die Phase des Allreduce zu sehen. Nachdem der erste Prozess fertig ist, sammelt er dann die Informationen der anderen Prozesse ein. Auch ist zu erkennen, dass die anderen Prozesse auf das Verteilen der Informationen des nullten Prozesses warten und danach erfolgt noch eine kleine weitere Berechnung auf jedem Prozess.

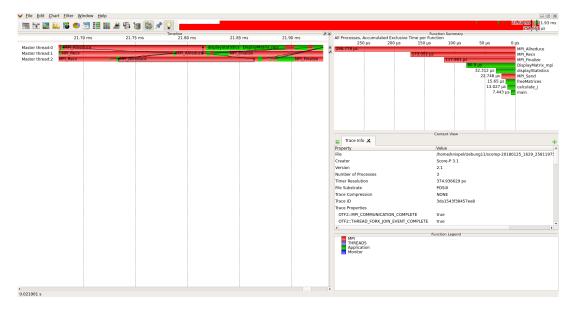


Abbildung 3: Jacobi-Verfahren mit 3 Prozessoren und 2 Knoten : Phase des Einsammelns

# 1.2 Gauß-Seidel-Verfahren

# 1.2.1 Startphase des Programms

Die Startphase ist so wie bei Jacobi. Nach der Initialisierung warten die Prozesse auf den Prozess darüber.

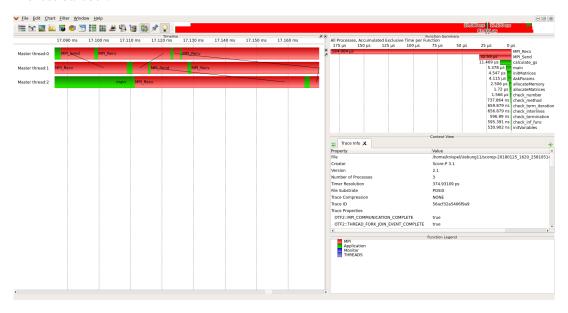


Abbildung 4: Gauß-Seidel-Verfahren mit 3 Prozessoren und 2 Knoten : Startphase

# 1.2.2 Phase der Synchronisation

Auch hier ist das gleiche Spiel wie bei Jacobi zu sehen, nur dass die aktuelle Iteration unterschiedlich ist.

#### 1.2.3 Phase des Einsammelns

Bei dem Allreduce wird wieder durch den nullten Prozess eingesammelt (kann man erkennen) und er verteilt danach die Informationen an die folgenden Prozesse. Da die anderen Prozesse dadurch ihre Information erhalten, wie viele Iterationen sie noch rechnen müssen, folgt darauf wieder ein Block von Berechnungen. Danach beenden auch hier alle Prozesse und Prozess null leitet die Funktion display-matrix ein.

# 2 Verwendung von 5 Prozessen und 4 Knoten

Bei der Verwendung von 5 Prozessen und 4 Knoten kann man die gleiche Vorgehensweise erkennen wie bei den Einstellungen darüber. Daher folgt hier keine weitere Erklärung dazu.

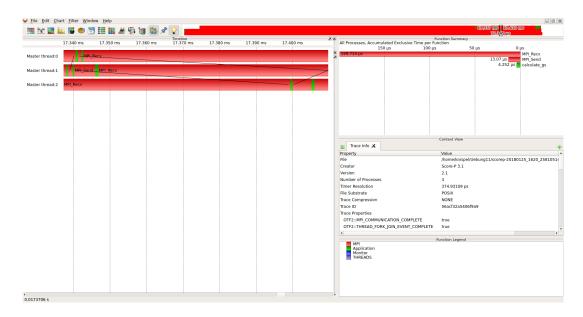


Abbildung 5: Gauß-Seidel-Verfahren mit 3 Prozessoren und 2 Knoten : Phase der Synchronisation

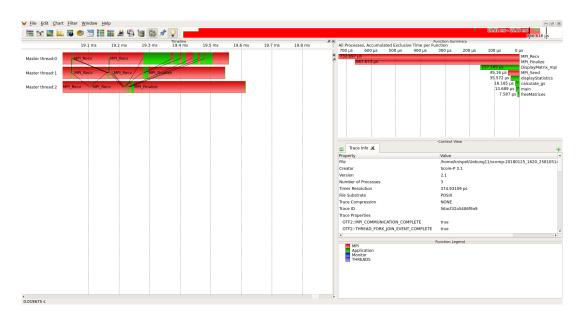


Abbildung 6: Gauß-Seidel-Verfahren mit 3 Prozessoren und 2 Knoten : Phase des Einsammelns

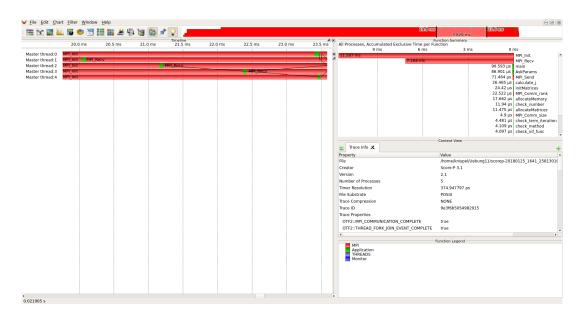


Abbildung 7: Jacobi-Verfahren mit 5 Prozessoren und 4 Knoten : Startphase

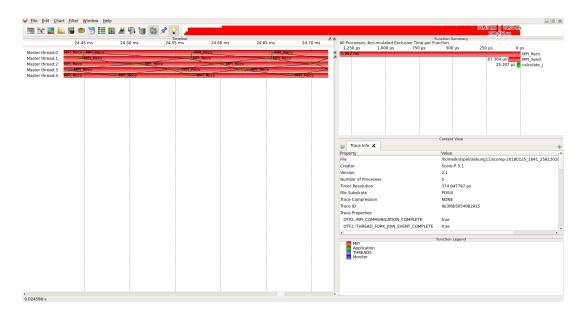


Abbildung 8: Jacobi-Verfahren mit 5 Prozessoren und 4 Knoten : Phase der Synchronisation

- 2.1 Jacobi-Verfahren
- 2.1.1 Startphase des Programms
- 2.1.2 Phase der Synchronisation
- 2.1.3 Phase des Einsammelns

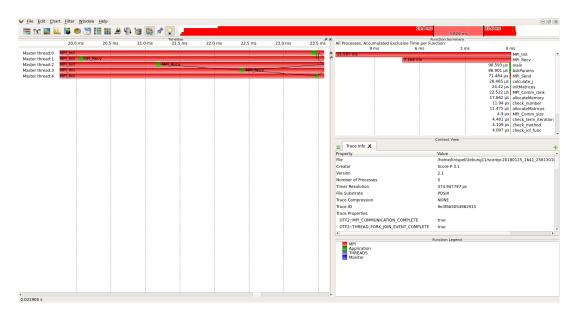


Abbildung 9: Jacobi-Verfahren mit 5 Prozessoren und 4 Knoten : Phase des Einsammelns

- 2.2 Gauß-Seidel-Verfahren
- 2.2.1 Startphase des Programms
- 2.2.2 Phase der Synchronisation
- 2.2.3 Phase des Einsammelns

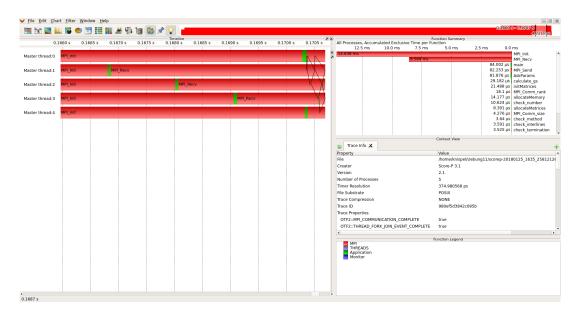


Abbildung 10: Gauß-Seidel-Verfahren mit 5 Prozessoren und 4 Knoten : Startphase

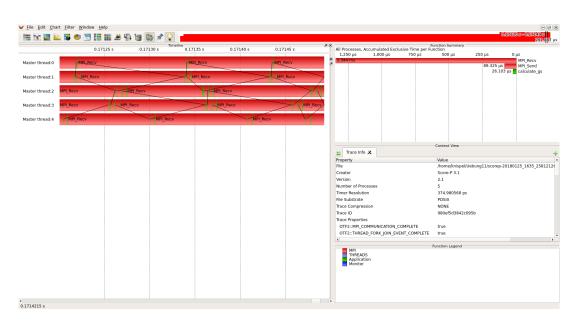


Abbildung 11: Gauß-Seidel-Verfahren mit 5 Prozessoren und 4 Knoten : Phase der Synchronisation

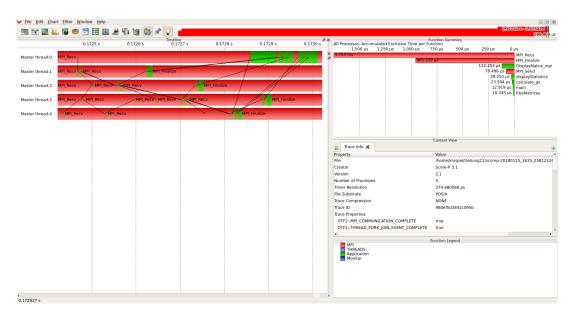


Abbildung 12: Gauß-Seidel-Verfahren mit 5 Prozessoren und 4 Knoten : Phase des Einsammelns