Aufgabe 2: Visualisierung

Es wurde ein Programmaufruf mit einem hohen N und 12 Prozessen mit scorep ausgewertet. Ein Aufruf von Vampir generierte dann eine grafische Darstellung des Programmablaufes. In der Zeitleiste am oberen Bildschirmrand wurde der interessante Teil ausgewählt (siehe Kreis) und in Bild 1 dargestellt. Links unterhalb sieht man das “timeline”-Fenster, in dem die Bereiche Initialisierung, Iterationen und Beenden markiert sind.

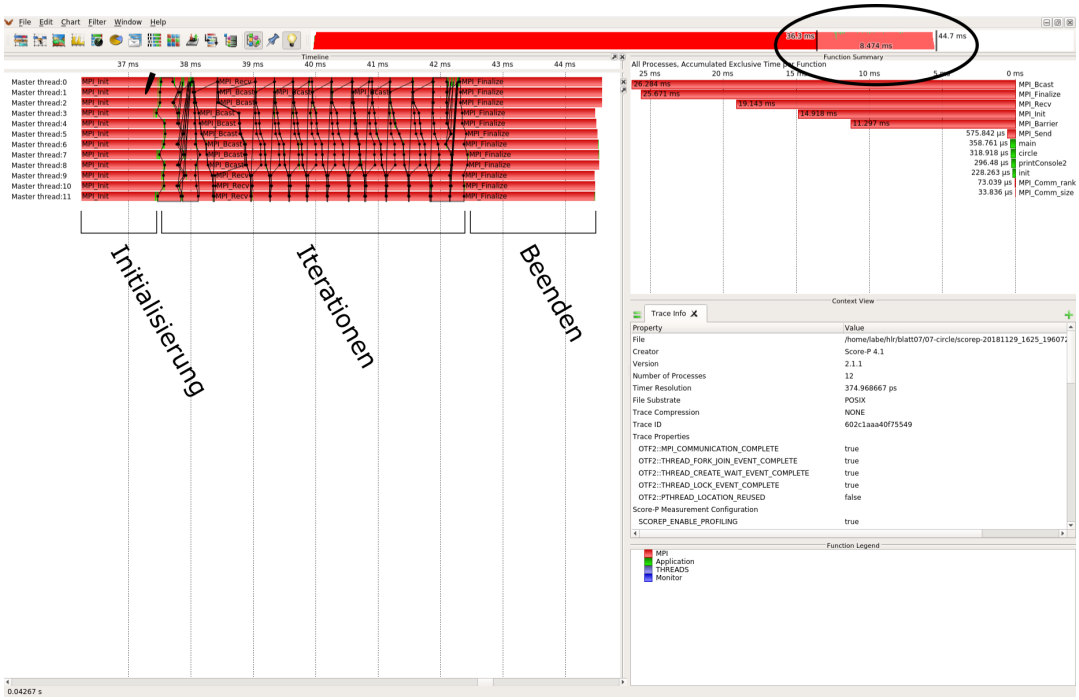

Figure 1: Vampir Übersicht

Wichtig ist hierbei zu bemerken, dass die Initialisierung deutlich länger braucht als zu sehen, das Diagramm beginnt durch die Wahl des Ausschnitts erst bei 36 ms.

Prozesse werden zeilenweise dargestellt, wobei Abschnitte die Ausführung verschiedener MPI-Befehle durch Abschnitte in den Balken gekennzeichnet ist (und dann mit Beschriftungen versehen ist, wie etwa MPI_Init ganz links). Mit schwarzen Linien werden dann die Kommunikationen zwischen den verschiedenen Prozessen dargestellt. Ein weiterer Zooom in den interessanten Teil des Programmes (die Iterationen) zeigt folgendes Bild:

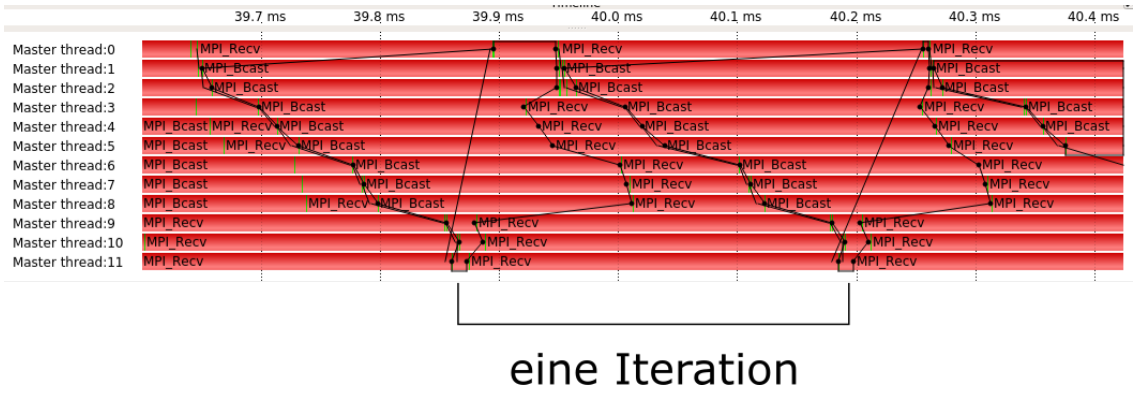

Figure 2: Ausschnitt aus den Iterationen

Hierbei wurde eine einzelne Iteration markiert. Die Kommunikation besteht aus zwei Hauptteilen, MPI_Recv (und MPI_Send) sowie MPI_BCAST. Wenn man die Kommunikationen von MPI_BCast aus dem Diagramm herausfiltert, erhält man folgendes:

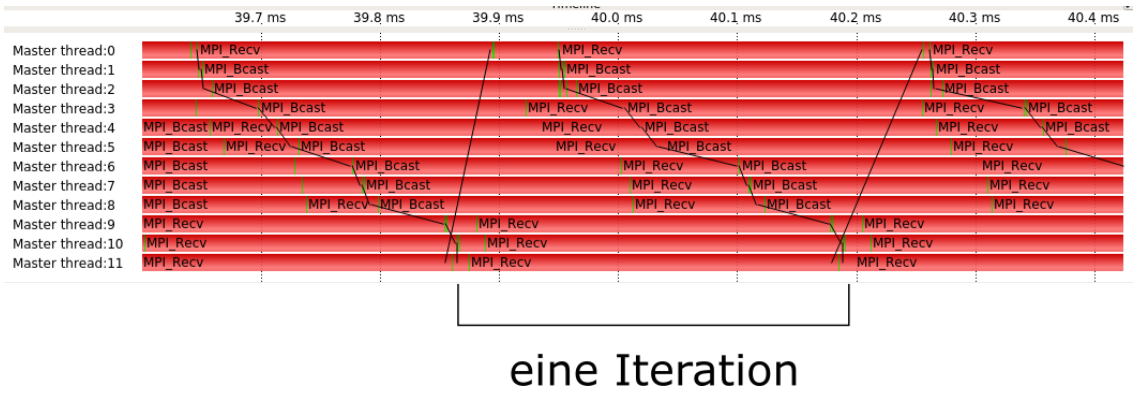

Figure 3: Ausschnitt aus den Iterationen, MPI_Bcast-Kommunikationen wurden herausgefiltert.

Es bleibt nur die Kommunitation aus MPI_Send und MPI_Recv stehen. Man sieht deutlich das erwartete Verhalten nach der Implementation in unserem circle.c: Prozess 0 schickt an 1, dieser an 2 usw. Da in allen Prozessen (außer 0) der MPI_Recv vor dem MPI_Send aufgerufen wird, erhält man das zu beobachtende Kaskadenartige Verhalten. Zum Schluss erhennt man deutlich die letzte Operation, in der Prozess 11 seine Daten an Prozess 0 schickt.

In 2 sieht man dann zusätzlich noch die Kommunikation von MPI_Bcast. Diese erfolgt zum einen entlang der Send-Recieve-Operationen, und zum anderen “aufwärts” laufend vom Prozess 11, welcher der “sendende” Prozess ist.