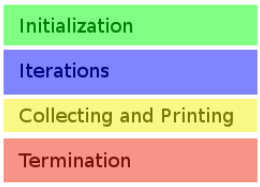
1 Anwendungsbeispiel

mpirun -np 8 circle 20

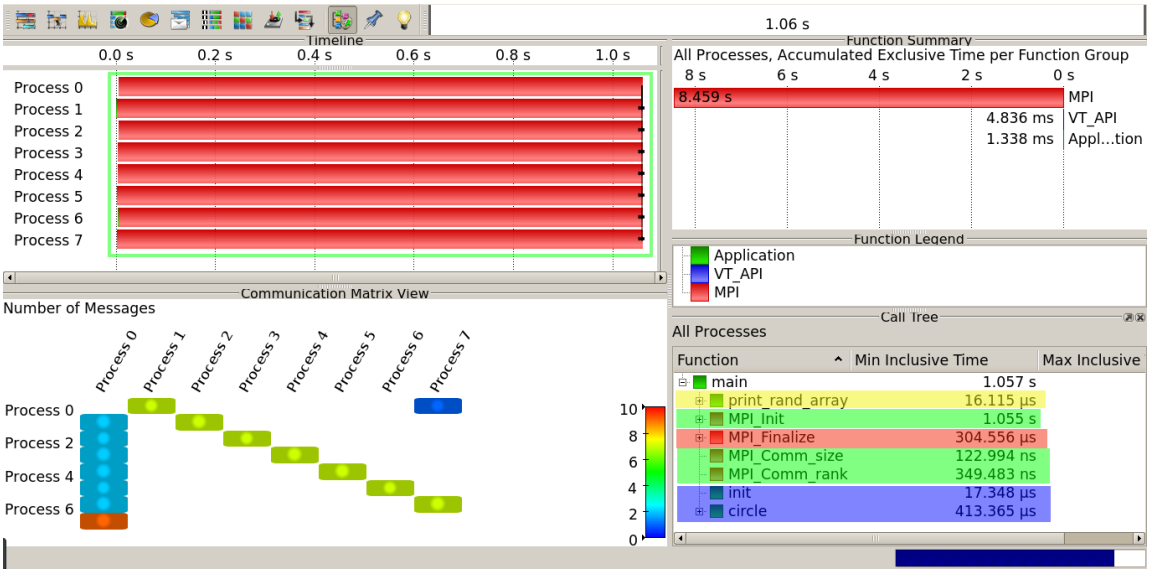
Before	After
rank 0: 22	rank 0: 1
rank 0: 12	rank 0: 5
rank 0: 11	rank 0: 7
rank 1: 1	rank 1: 13
rank 1: 5	rank 1: 6
rank 1: 7	rank 1: 23
rank 2: 13	rank 2: 14
rank 2: 6	rank 2: 6
rank 2: 23	rank 2: 3
rank 3: 14	rank 3: 1
rank 3: 6	rank 3: 14
rank 3: 3	rank 4: 20
rank 4: 1	rank 4: 5
rank 4: 14	rank 5: 1
rank 5: 20	rank 5: 11
rank 5: 5	rank 6: 11
rank 6: 1	rank 6: 5
rank 6: 11	rank 7: 22
rank 7: 11	rank 7: 12
rank 7: 5	rank 7: 11

2 Visualisierung

Die folgende Farbkodierung gilt für alle Screenshots:

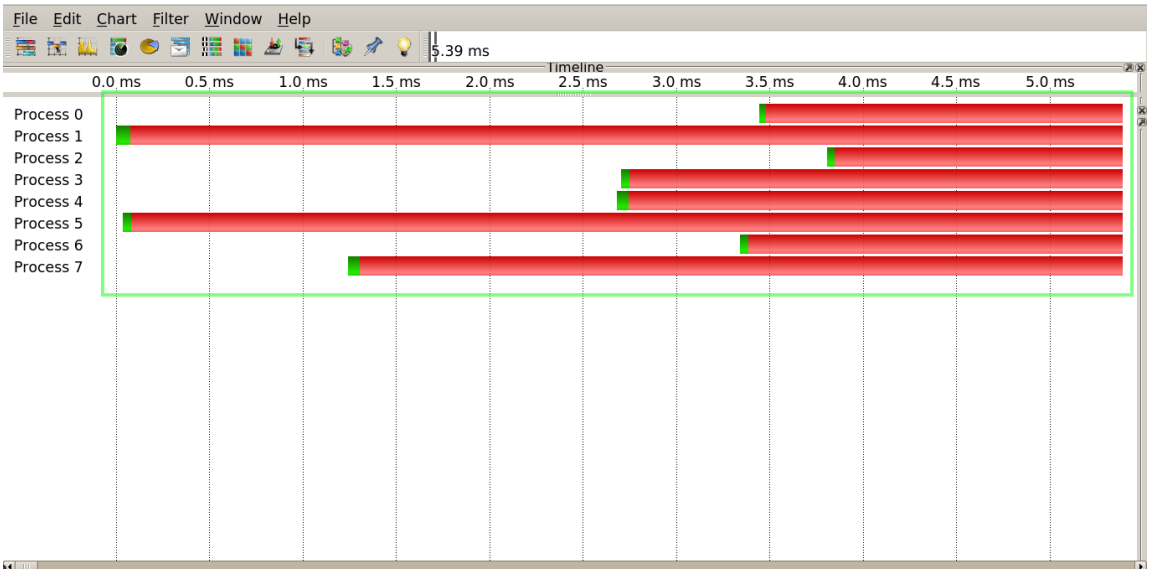


2. a)



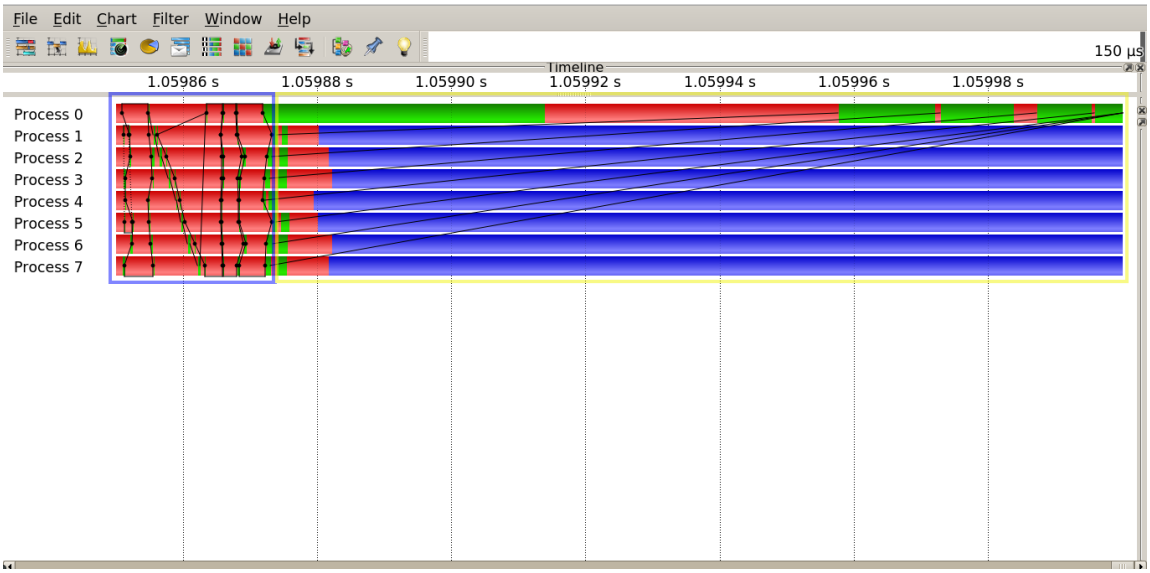
Links oben, in dem grünen Kasten ist dargestellt, zu welcher Zeit sich jeder Prozess innerhalb von MPI-Funktionen, VT_API-Funktionen oder programmeigenen Funktionen aufhielt. Deutlich erkennbar ist dabei, dass der Parallelisierungsoverhead (v.a. Initialisierung) in diesem Fall wesentlich mehr Zeit in Anspruch nahm, als der eigentliche Programmcode (Das berechnen, Vergleichen und Ausgeben der Zufallsvariablen).
Rechts oben ist dasselbe noch einmal für das Programm insgesamt aufgeführt.
Links unten ist in einer Matrix dargestellt, wie oft jedes Paar von Prozessen miteinander kommuniziert hat. Erkennbar ist, dass jeder Prozess mit dem letzten kommuniziert hat, da dieser den anderen einmal pro Iteration die Abbruchvariable mitgeteilt hat. Ansonsten hat jeder Prozess nur je einmal mit seinem Vorgänger und Nachfolger kommuniziert.
Rechts unten ist die Zeit aufgeführt, die das Programm in jeder Funktion verbracht hat. Ich habe farbig markiert, welche Funktion zu welcher Programmphase gehört.

2. b)



Hier ist zu sehen, wann die einzelnen Prozesse initialisiert wurden. Dies findet offensichtlich in der Initialisierungsphase statt.

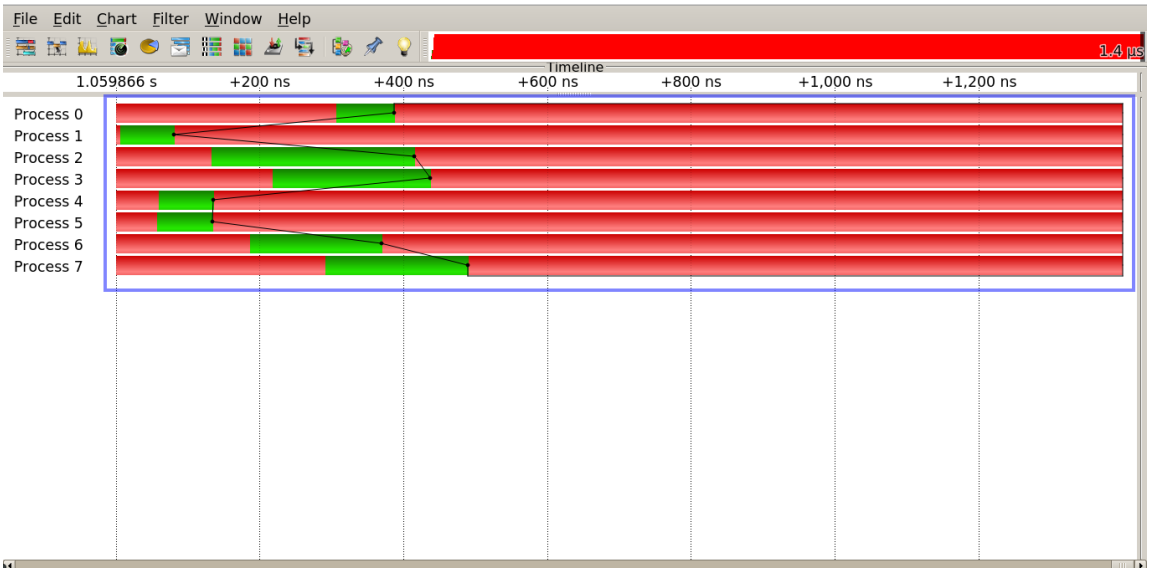
2. c)



Hier ist dargestellt, zu welcher Zeit die Prozesse arbeiteten (grün), von MPI blockiert waren (rot) oder schon terminiert hatten (blau). Im Bereich innerhalb des blauen Kastens befindet sich das Programm in der while-Schleife während welcher die Zufallsvariablen ausgetauscht werden.

Im Bereich innerhalb des gelben Kastens (evtl. schwer zu sehen) befindet sich das Programm in der Ausgabephase (alle Prozesse außer dem Masterprozess haben bereits terminiert).

2. d)



Hier ist eine Iteration der while-Schleife zu sehen. Die durch die schwarze Linie verbundenen Punkte markieren die Zeitpunkte an denen jeder Prozess die Barrier erreicht hat.