

Aufgabe 1) Ein Programm ist eine Sammlung von Anweisungen/Befehlen in Form einer Datei als Code. Das Programm

beschreibt dabei, was der Computer tun soll, sobald dies ausgeführt wird. Ohne Kompilierung bzw. Ausführung der Datei, werden die Anweisungen nicht ausgeführt. Ein Prozess hingegen, ist dabei ein aktiver, instanzierter Zustand des Programms. Dieser wird erstellt, wenn ein Programm ausgeführt wird. Dieser enthält alle Informationen wie bspw. benötigte Ressourcen, Speicherverwaltung (\rightarrow Zuweisung) usw.

Sollten allerdings mehrere Prozesse gleichzeitig laufen, dann arbeiten diese unabhängig voneinander.

Als letztes der Thread, der innerhalb eines Prozesses existiert. Er bildet die kleinste Einheit eines Programms bzw. einer Ausführung.

Diese verarbeiten letztlich alle Informationen innerhalb eines Programms auf dem Computer. Der erste erzeugte Hauptthread, ist der main-Thread.

Ebenso kann ein Prozess mehrere Threads erzeugen (Multithreading). Allerdings arbeiten alle mit den selben Ressourcen innerhalb des Prozesses.

Dazu muss eine effiziente Kommunikation zwischen ihnen existieren, da sonst Sicherheitsprobleme auftreten können oder es bspw. zur Datenkorruption führen könnte.

Aufgabe 2) Speedup bei $\overbrace{25, 50 \text{ & } 75\%}^{\text{parallelisierbarer Teil}}$: für 2^x Kerne mit $0 \leq x \leq 3$

$$75\%: s = 0,25 \quad (1-s) = 0,75 \quad \text{Single Core: } t_{\text{exe}} = 1s$$

parallelisierbar

$$2\text{-Core: } 0,25 + \frac{0,75}{2} = 0,625 \Rightarrow t_{\text{exe}} = 0,625s \quad \text{Speedup: } \frac{1}{0,625} =$$

$$4\text{-Core: } 0,25 + \frac{0,75}{4} = 0,4375 \Rightarrow t_{\text{exe}} = 0,4375 \quad \text{Speedup: } \frac{1}{0,4375} \approx 2,26$$

$$8\text{-Core: } 0,25 + \frac{0,75}{8} = 0,34375 \Rightarrow t_{\text{exe}} = 0,34375 \quad \text{Speedup: } \frac{1}{0,34375} \approx 2,909$$

$$50\%: s = 0,5 \quad (1-s) = 0,5 \quad \text{Single Core: } 0,5 + 0,5 = 1s \quad t_{\text{exe}} = 1s \quad \text{Speedup: } 1$$

parallelisierbar

$$2\text{-Core: } 0,5 + \frac{0,5}{2} = 0,75 \quad t_{\text{exe}} = 0,75 \quad \text{Speedup: } \frac{1}{0,75} = 1\frac{1}{3}$$

$$4\text{-Core: } 0,5 + \frac{0,5}{4} = 0,625 \quad t_{\text{exe}} = 0,625 \quad \text{Speedup: } \frac{1}{0,625} = 1,6$$

$$8\text{-Core: } 0,5 + \frac{0,5}{8} \approx 0,5625 \quad t_{\text{exe}} = 0,5625 \quad \text{Speedup: } \frac{1}{0,5625} = 1,7$$

$$25\%: s = 0,75 \quad (1-s) = 0,25 \quad \text{Single Core: } t_{\text{exe}} = 1s \quad \text{Speedup: } 1$$

parallelisierbar

$$2\text{-Core: } 0,75 + \frac{0,25}{2} = 0,875 \quad t_{\text{exe}} = 0,875s \quad \text{Speedup: } \frac{1}{0,875} = \frac{8}{7}$$

$$4\text{-Core: } 0,75 + \frac{0,25}{4} = \frac{10}{12} = \frac{5}{6}s \quad t_{\text{exe}} = 0,83s \quad \text{Speedup: } \frac{1}{0,83} = \frac{6}{5} = 1,2$$

$$8\text{-Core: } 0,75 + \frac{0,25}{8} = 0,78125s \quad t_{\text{exe}} = 0,78125 \quad \text{Speedup: } \frac{1}{0,78125} = 1,28$$

Aufgabe 3)

Die veränderte Datei für Multithreading lautet: "Counter-Multithread"

Über ein drittes Argument können wir unsere Anzahl an Threads festlegen, welche wir mit ForkJoinPool kontrollieren.

Diese werden an passender Stelle (nach Filterung) mit parallel() parallelisiert.

Verzeichnis nach .txt-Dateien durchsucht

Anwendungstest: Anweisung: java Counter-Multithread E:\Uni *.txt n (mit n als Platzhalter für Anzahl der Threads)

Threads	Test 1	Test 2
1	Count Lines → 76131 Time Taken → 0,326 sec	Count Lines → 76131 Time Taken → 0,303 sec
2	Count Lines → 76131 Time Taken → 0,247 sec	Count Lines → 76131 Time Taken → 0,25 sec
4	Count Lines → 76131 Time Taken → 0,224	Count Lines → 76131 Time Taken → 0,244
8	Count Lines → 76131 Time Taken → 0,225	Count Lines → 76131 Time Taken → 0,234
16	Count Lines → 76131 Time Taken → 0,253	Count Lines → 76131 Time Taken → 0,248
32	Count Lines → 76131 Time Taken → 0,253	Count Lines → 76131 Time Taken → 0,237

Da das Verzeichnis E:\Uni nicht allzu groß ist haben wir auch nur 76131 counted Lines.

Ebenso sieht man, dass es bis zu 4 Threads deutliche Verbesserungen gab. Allerdings ist für alles mit mehr als 4 Threads keine eigentliche

Verbesserung ersichtlich. Dies liegt an der geringen Mächtigkeit des Verzeichnisses.