**Gruppe 15 | Tobias Schoch, Luis Nothvogel**

**Aufgaben wurden auf dem HTWG Container ausgeführt**

**29.4**

Die Vermutung liegt nahe, dass bei geringer Threadanzahl die HoH-LinkedList schlechter performed als die normale LinkedList denn für wenige Iterationen wäre dies zu viel Overhead. Bei höherer Threadanzahl performed vermutlich die HoH-LinkedList besser, da gleichzeitig drüber iteriert werden kann. Und bei der LinkedList muss jedes mal auf die gesamte Iteration gewartet werden.

Im Beispiel unten haben wir jeweils die Version aus dem Buch (linkedlist) und unsere (linkedListHoH) mit 1 Thread und mit 20 Threads laufen lassen. Wir haben hierbei eine Listgröße von 10000 verwendet und das zu findende Element 60. Wir haben verschieden große Listgrößen zum testen verwendet (10, 100, 1000, 10000, 100000). Bei kleineren Listen (10, 100, 100, 1000) war die Version aus dem Buch immer schneller egal wie viele Threads wir hatten. Erst bei größeren Listgrößen von (10.000, 100.000…) ergab sich das angenommene Verhalten. Also ist anzunehmen, dass man erst eine bestimmte Anzahl von Iteration benötigt damit die HoH-LinkedList besser ist.

Bsp.





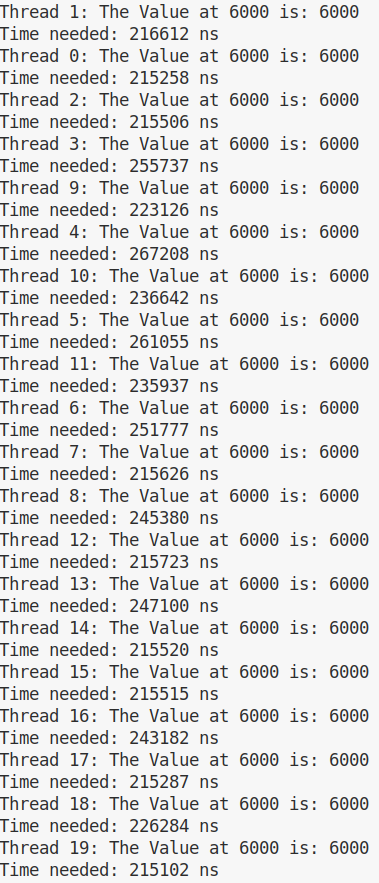
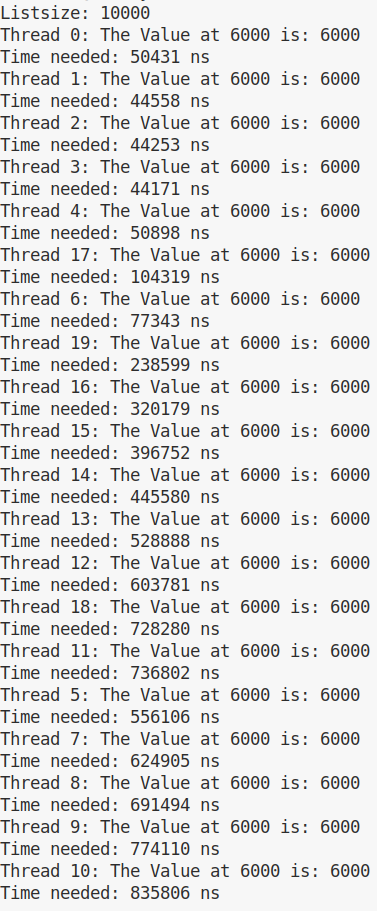


Abbildung ohne HoH

Abbildung HoH

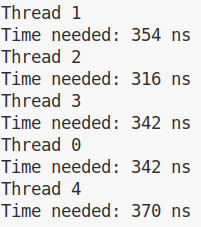
**29.5**

Wir haben uns für einen balancierten Binärbaum entschieden mit einfachem Lock.

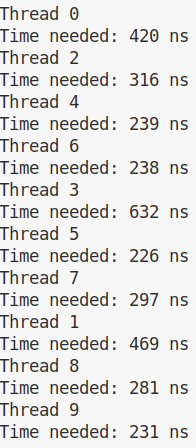




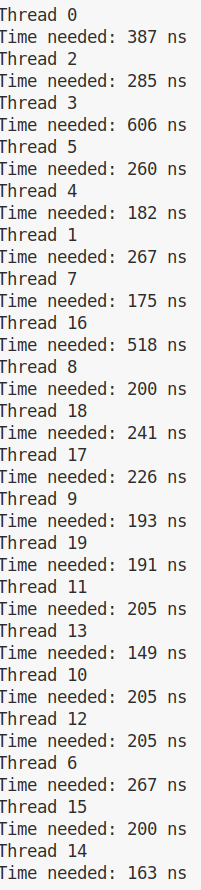












Wie man an den Screenshots sehen kann, wird die Performance immer besser je mehr Threads dazu kommen. Allerdings haben wir bei 20 Threads einen Performance Cap. Bei 30, 40 … usw. wurde es nicht mehr besser von der Performance.

**29.6**

Wir haben im Grunde das Gleiche gemacht wie bei der LinkedListe. Denn wir haben uns für ein Couple Locking entschieden. Die Erwartung wird vermutlich ähnlich wie bei der LinkedList sein. D.h. bei geringer Threadanzahl dürfte die Version mit dem simplen Lock besser performen und bei höherer Threadanzahl dürfte die Version mit couple locking besser performen.

Wir haben das ganze wie bei 29.5 mit verschieden vielen Threads getestet. Wir haben immer 600 Iterationen verwendet. Allerdings war zu bemerken, dass die Version mit dem simplen lock immer schneller war. Wir haben das ganze dann nochmal mit mehr Iteration wie bei 29.4 getestet, doch es kam immer noch dasselbe raus.





