Take Home Lab 1

## Group Members

210113L - Akindu Delgahagoda  
210195L - Induwara Gunasena

## Step 3

### Case 1

n = 1,000 and m = 10,000, mMember = 0.99, mIndert = 0.005, mDelete = 0.005

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Implementation | No of threads | | | | | | | |
| 1 | | 2 | | 4 | | 8 | |
| Avg | Std | Avg | Std | Avg | Std | Avg | Std |
| Serial | 7053 | 949 |  |  |  |  |  |  |
| One mutex for entire list | 8135 | 702 | 13112 | 1210 | 14710 | 896 | 14597 | 490 |
| Read-Write lock | 8848 | 656 | 7598 | 421 | 6286 | 847 | 7223 | 727 |

### Case 2

n = 1,000 and m = 10,000, mMember = 0.90, mIndert = 0.05, mDelete = 0.05

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Implementation | No of threads | | | | | | | |
| 1 | | 2 | | 4 | | 8 | |
| Avg | Std | Avg | Std | Avg | Std | Avg | Std |
| Serial | 10648 | 878 |  |  |  |  |  |  |
| One mutex for entire list | 11257 | 969 | 19073 | 1716 | 20240 | 1017 | 21313 | 1191 |
| Read-Write lock | 13350 | 3383 | 17502 | 707 | 19358 | 1188 | 19884 | 1026 |

### 

### Case 3

n = 1,000 and m = 10,000, mMember = 0.50, mIndert = 0.25, mDelete = 0.25

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Implementation | No of threads | | | | | | | |
| 1 | | 2 | | 4 | | 8 | |
| Avg | Std | Avg | Std | Avg | Std | Avg | Std |
| Serial | 44699 | 1511 |  |  |  |  |  |  |
| One mutex for entire list | 45773 | 1797 | 61931 | 3610 | 63750 | 1778 | 65038 | 1551 |
| Read-Write lock | 46052 | 1562 | 70865 | 2821 | 73178 | 1234 | 72717 | 1319 |

## Device Specifications

Brand – Asus VivoBook X513EP-BQ292T

Operating System - Windows 11 Home 64Bit

Processor - Intel Core i7

RAM - 16GB DDR4

Storage - 256 SSD + ITB HDD (Program was stored in HDD)

Here is the output of the *lscpu* command in WSL environment.

Architecture: x86\_64

CPU op-mode(s): 32-bit, 64-bit

Address sizes: 39 bits physical, 48 bits virtual

Byte Order: Little Endian

CPU(s): 4

On-line CPU(s) list: 0-3

Vendor ID: GenuineIntel

Model name: 11th Gen Intel(R) Core(TM) i7-1165G7 @ 2.80GHz

CPU family: 6

Model: 140

Thread(s) per core: 2

Core(s) per socket: 2

Socket(s): 1

Stepping: 1

BogoMIPS: 5606.40

Flags: fpu vme de pse tsc msr pae mce cx8 apic sep mtrr pge mca cmov pat pse3

6 clflush mmx fxsr sse sse2 ss ht syscall nx pdpe1gb rdtscp lm constan

t\_tsc arch\_perfmon rep\_good nopl xtopology tsc\_reliable nonstop\_tsc cp

uid tsc\_known\_freq pni pclmulqdq vmx ssse3 fma cx16 pdcm pcid sse4\_1 s

se4\_2 x2apic movbe popcnt tsc\_deadline\_timer aes xsave avx f16c rdrand

hypervisor lahf\_lm abm 3dnowprefetch ssbd ibrs ibpb stibp ibrs\_enhanc

ed tpr\_shadow ept vpid ept\_ad fsgsbase tsc\_adjust bmi1 avx2 smep bmi2

erms invpcid avx512f avx512dq rdseed adx smap avx512ifma clflushopt cl

wb avx512cd sha\_ni avx512bw avx512vl xsaveopt xsavec xgetbv1 xsaves vn

mi avx512vbmi umip avx512\_vbmi2 gfni vaes vpclmulqdq avx512\_vnni avx51

2\_bitalg avx512\_vpopcntdq rdpid movdiri movdir64b fsrm avx512\_vp2inter

sect md\_clear flush\_l1d arch\_capabilities

Virtualization features:

Virtualization: VT-x

Hypervisor vendor: Microsoft

Virtualization type: full

Caches (sum of all):

L1d: 96 KiB (2 instances)

L1i: 64 KiB (2 instances)

L2: 2.5 MiB (2 instances)

L3: 12 MiB (1 instance)

NUMA:

NUMA node(s): 1

NUMA node0 CPU(s): 0-3

Vulnerabilities:

Gather data sampling: Unknown: Dependent on hypervisor status

Itlb multihit: Not affected

L1tf: Not affected

Mds: Not affected

Meltdown: Not affected

Mmio stale data: Not affected

Reg file data sampling: Not affected

Retbleed: Mitigation; Enhanced IBRS

Spec rstack overflow: Not affected

Spec store bypass: Mitigation; Speculative Store Bypass disabled via prctl

Spectre v1: Mitigation; usercopy/swapgs barriers and \_\_user pointer sanitization

Spectre v2: Mitigation; Enhanced / Automatic IBRS; IBPB conditional; RSB filling;

PBRSB-eIBRS SW sequence; BHI SW loop, KVM SW loop

Srbds: Not affected

Tsx async abort: Not affected

## Step 4

## Step 5

**Case 1 – Read-heavy workload (99% Member, 0.5% Insert/Delete):**In this scenario, the read-write lock implementation demonstrates a significant performance advantage over both the serial and single-mutex versions. Because most operations are reads, multiple threads can execute Member operations concurrently without blocking each other, resulting in lower execution times as thread count increases. The single-mutex approach shows only minor improvements since all operations, including reads, must acquire the same lock, which limits parallelism. Overall, read-heavy workloads benefit greatly from read-write locks, achieving both speedup and scalability.

**Case 2 – Mixed workload (90% Member, 5% Insert/Delete):**For a moderately read-heavy workload, read-write locks still outperform single-mutex implementations, but the improvement is less pronounced. The presence of write operations (Insert/Delete) introduces periods where exclusive access is required, blocking other threads and reducing the effectiveness of concurrent reads. Single-mutex performance remains limited, as all threads must acquire the same lock regardless of operation type. This case highlights that the benefit of read-write locks decreases as the fraction of writes increases, though some parallel efficiency is still observed.

**Case 3 – Balanced/write-heavy workload (50% Member, 25% Insert/Delete):**In a workload with a high proportion of writes, read-write locks offer little advantage over single-mutex implementation. Most operations require exclusive access, causing threads to wait frequently and limiting parallel speedup. Execution times for both parallel implementations are comparable, and increasing thread count does not significantly reduce the runtime. This demonstrates that read-write locks are most effective for read-dominated workloads and that their performance benefit diminishes as write operations become frequent. Standard deviation values reflect slight variability due to contention and thread scheduling.