Sharan Babu Paramasivam Murugesan (sxp141731)

Vasudev Karthik Ravindran (vxr143630)

Report – Project 3

Experimental setup:

Programming Language : Java
Operating system : Linux (CentOS distribution)
Hardware:
Processor - Dual 8-core Intel Xeon E5-2680 2.7Ghz
Type of locking – Coarse grained & Fine Grained Locking

The experiment is done with varying the following parameters –

Number of threads – 1, 2, 4, 8, 16, 32
Key Space - 1000, 10000, 100000, 1000000
Relative distribution – A, B, C where

A indicates read-dominated operation (90% search, 9% insert and 1% delete)
B indicates mixed operation (70% search, 20% insert and 10% delete)
C indicates write-dominated operation (0% search, 50% insert and 50% delete)

Each data point in the graphs are averaged over 10 runs

External libraries – none

Verification Procedure –

- The constructed BST with external representation is verified by checking if all the nodes are satisfying the BST property where in left child has lesser value than root as well as the right child has value greater than or equal to that of root.
- Also an in-order traversal is performed on the tree and the result output is checked if it is in sorted order
- Every node value in the final tree is checked for maximum occurrence of two times

Through put measured – Number of operations completed per 1 ms

Synchronization Algorithms used:

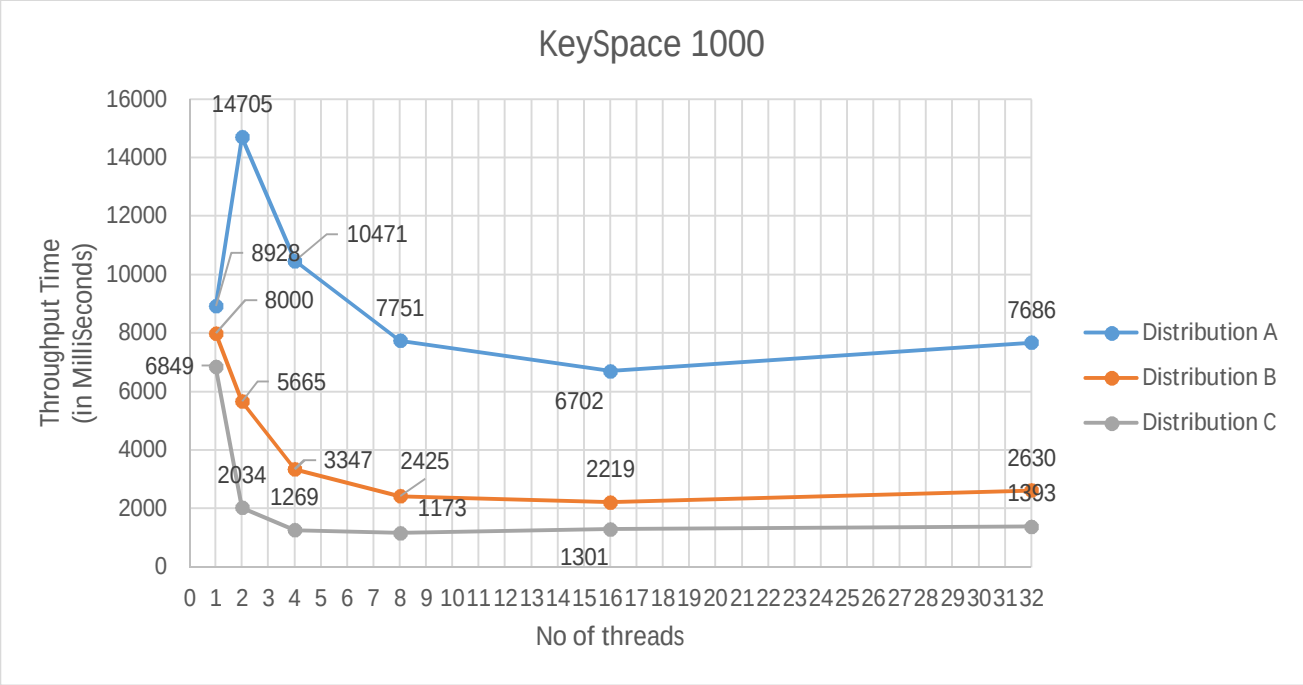
1. Coarse Grained locking
2. Fine-grained locking using Lazy Synchronization

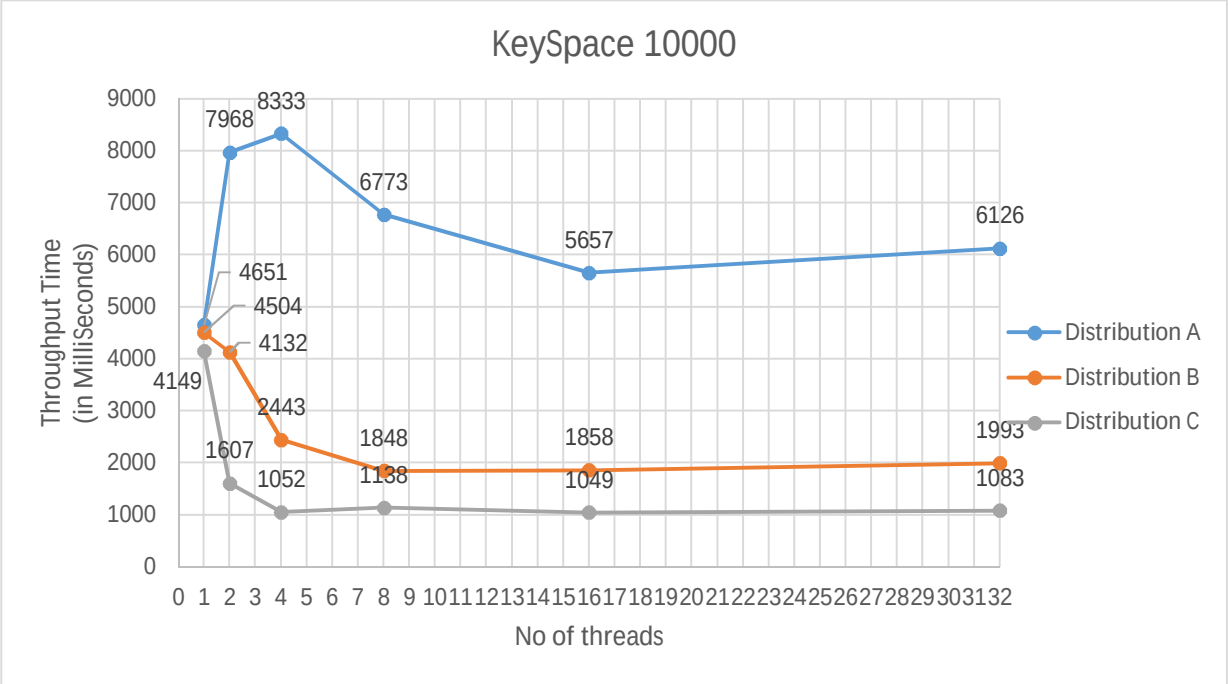
Results:

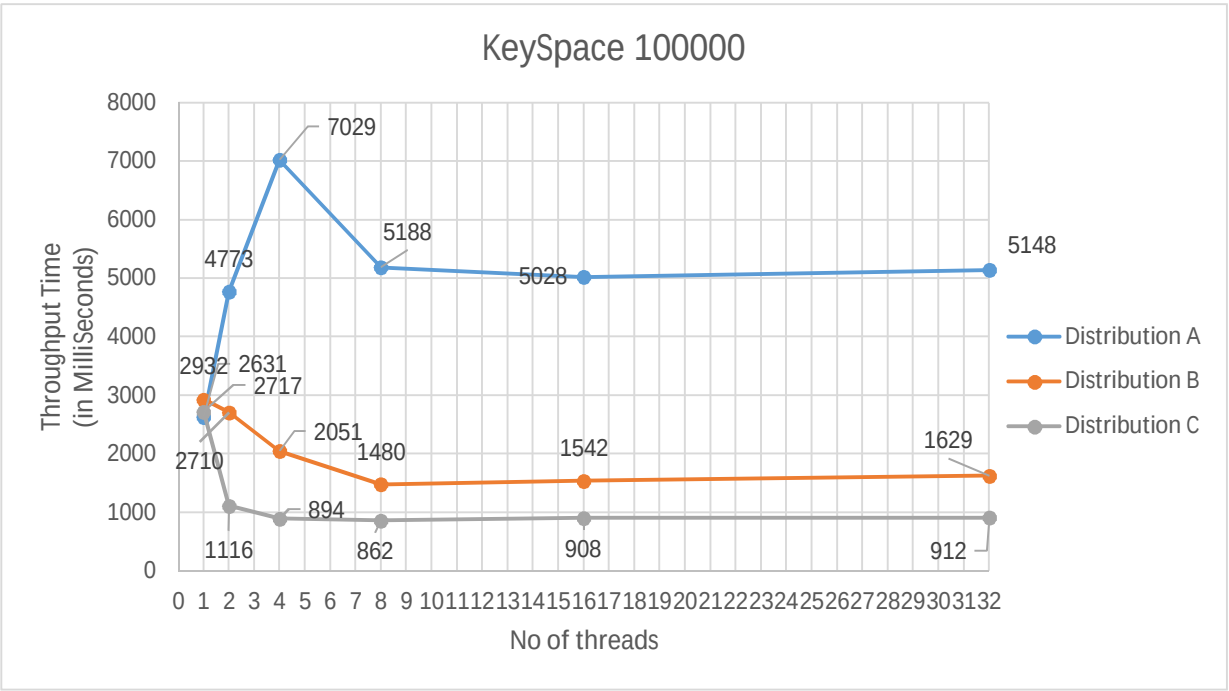
Each data point in the below shown results are averaged over 10 runs

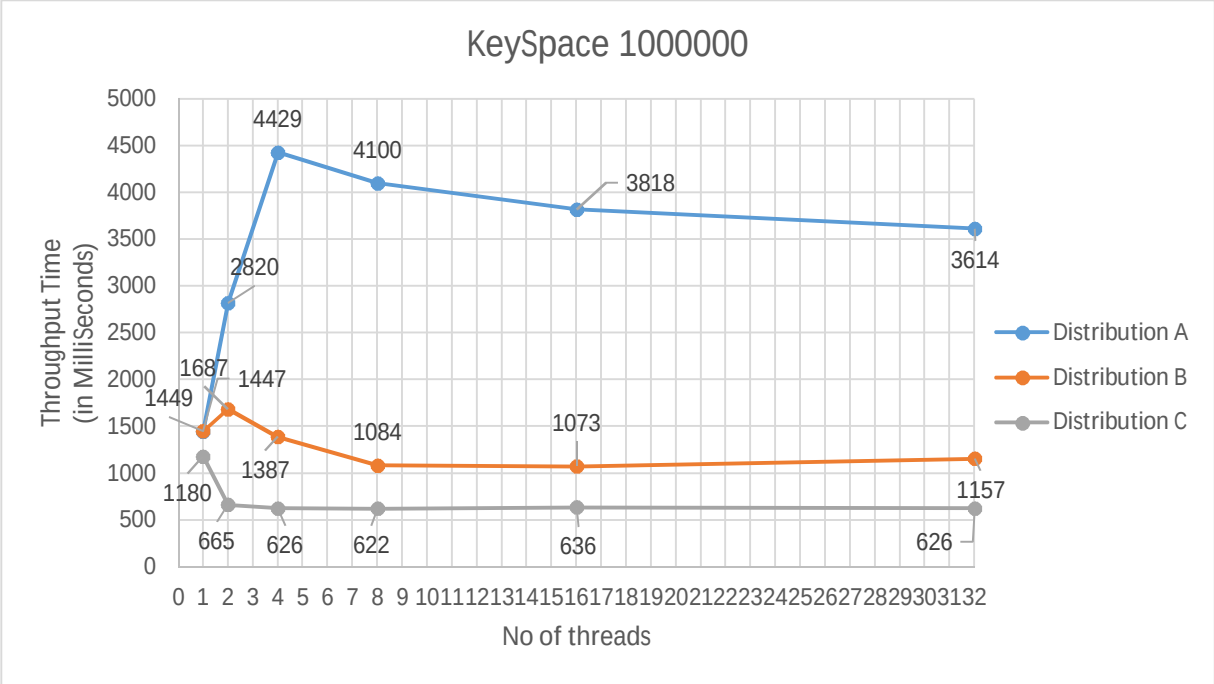
Coarse Grained Locking:

Below are the results for Coarse Grained Locking with each graph for different keyspaces:




KeySpace 10000
Throughput Time (in MilliSeconds)
No of threads
Distribution A
Distribution B
Distribution C
7968
8333
6773
5657
6126
4651
4504
4132
4149
2443
1848
1858
1993
1607
1052
1138
1049
1083

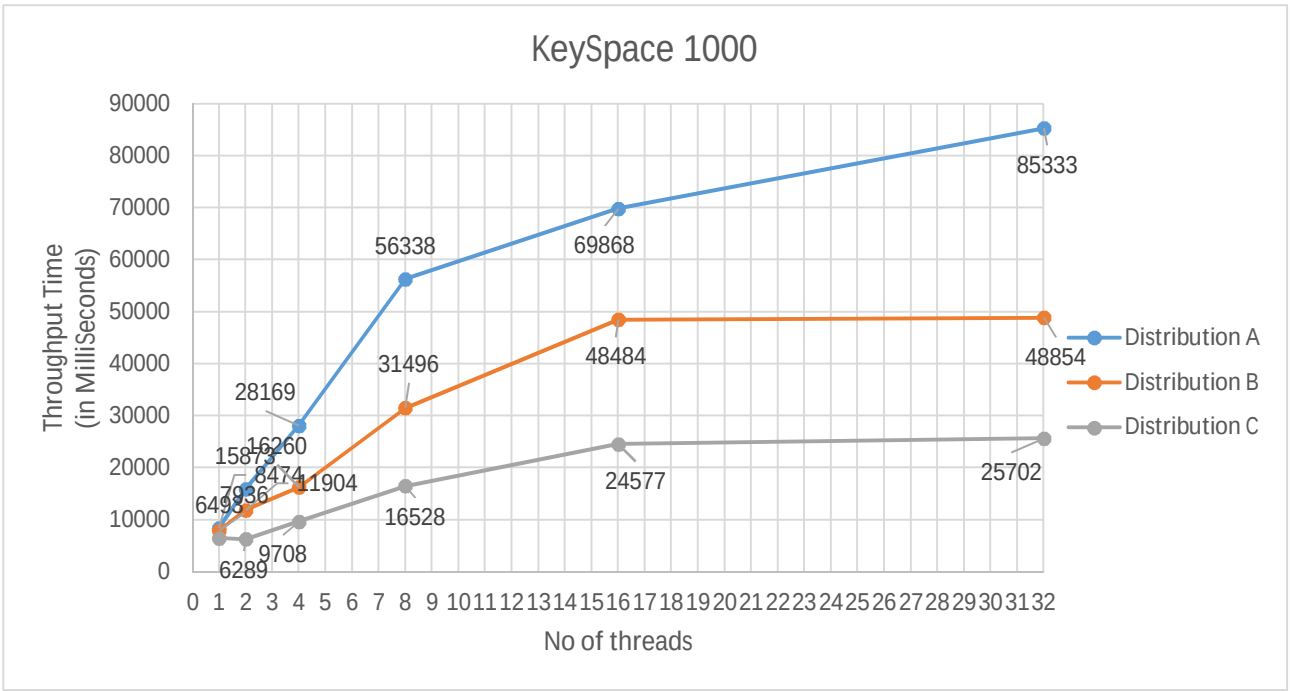

KeySpace 100000
Throughput Time (in MilliSeconds)
No of threads
Distribution A
Distribution B
Distribution C
7029
5188
4773
5028
5148
2932
2631
2717
2051
2710
1480
1542
1629
894
1116
862
908
912

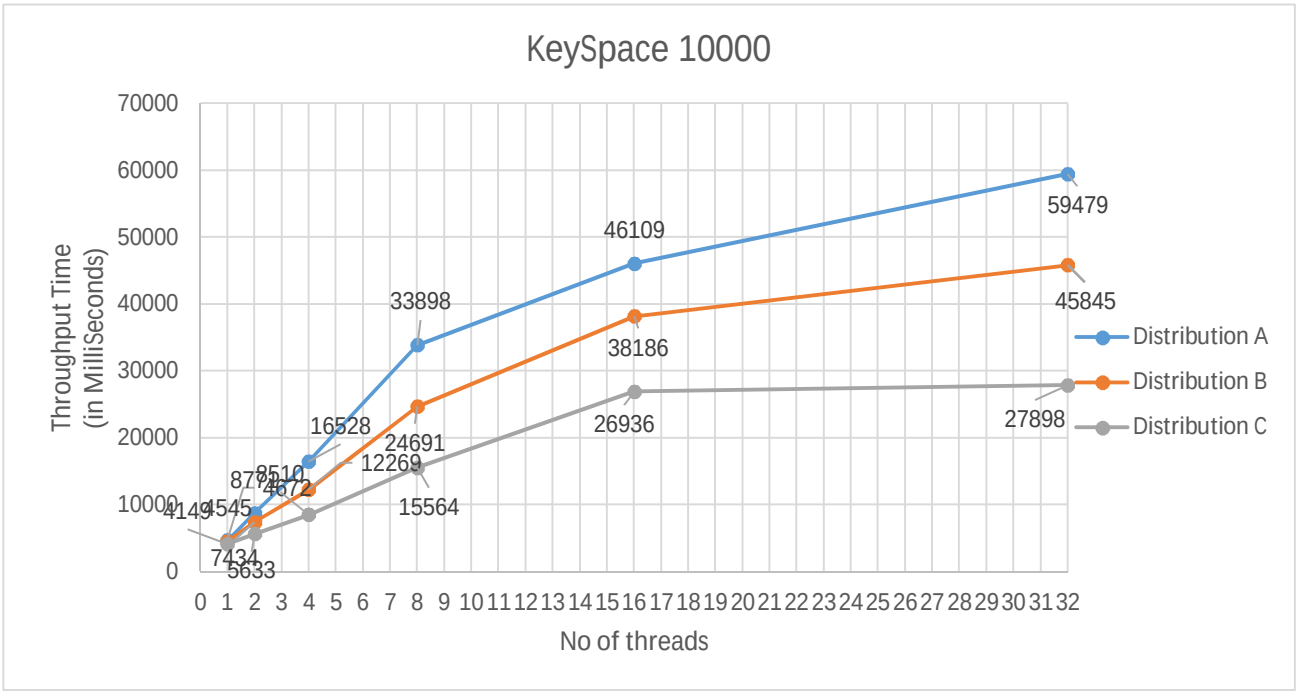


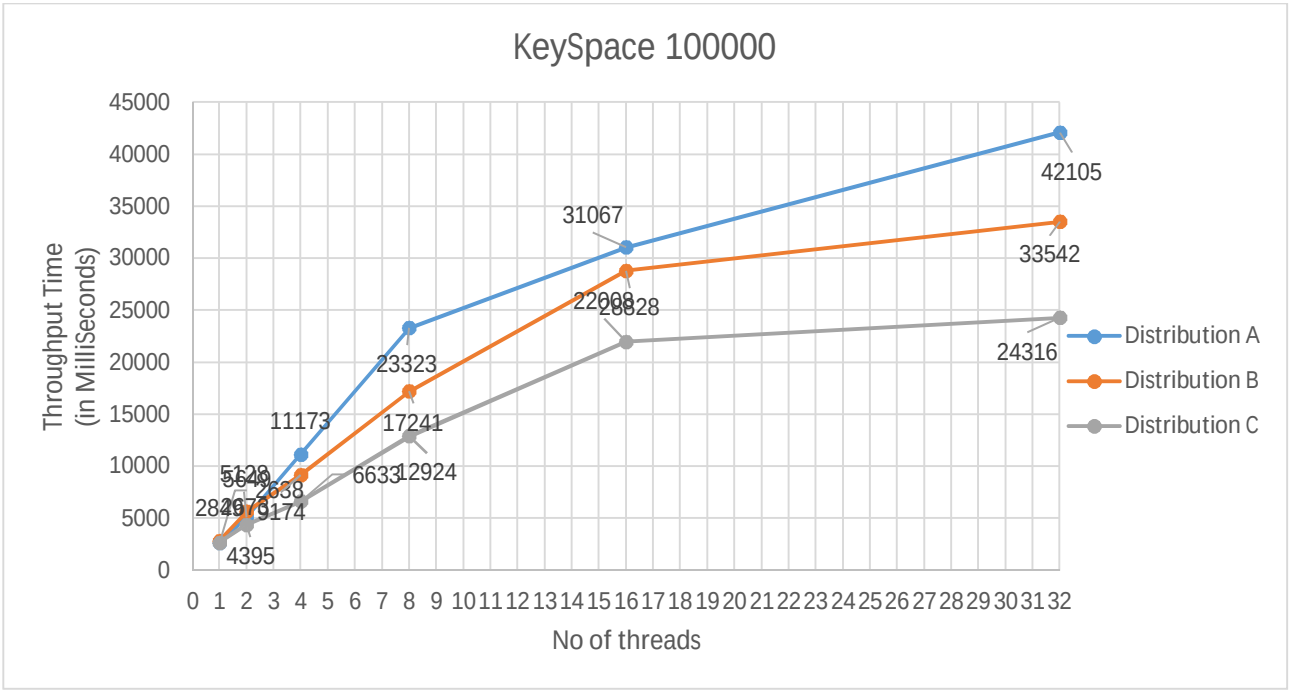
Fine Grained Locking:

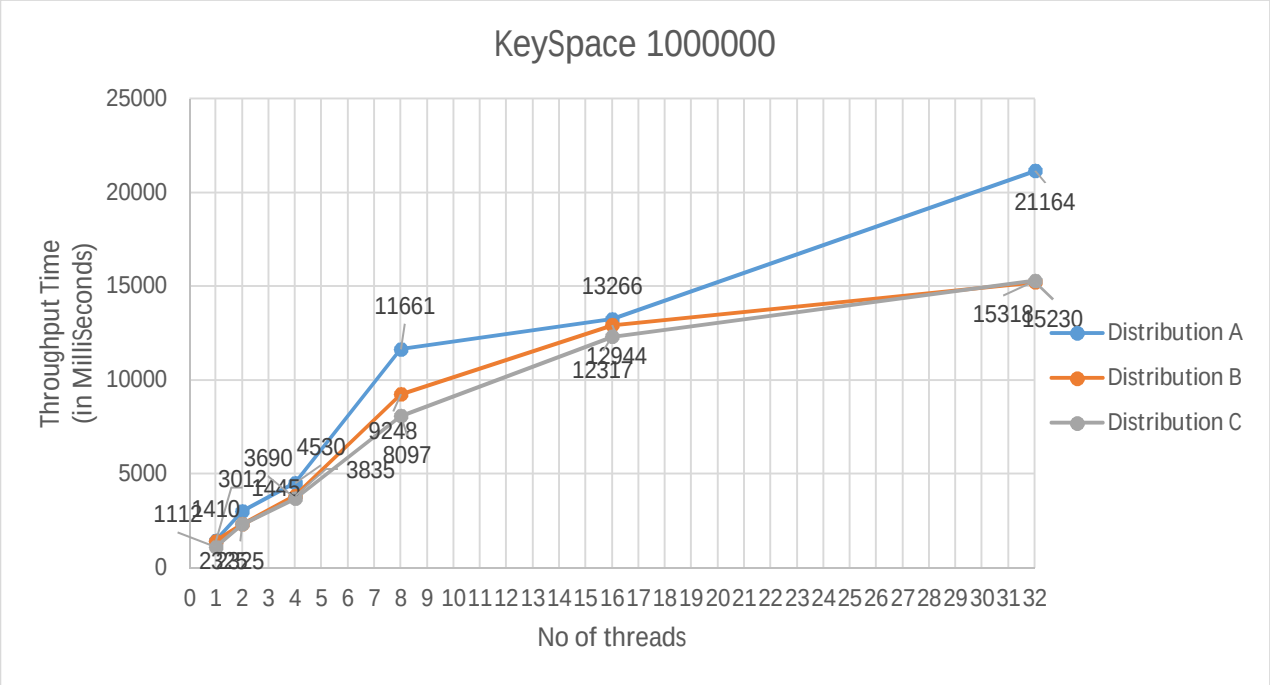
Below are the results for Fine Grained Locking with each graph for different keyspaces:

.




KeySpace 10000
Throughput Time (in MilliSeconds)
No of threads
Distribution A
Distribution B
Distribution C
59479
46109
33898
16528
45845
38186
24691
12269
27898
26936
15564
8510
8774
4672
4149
4545
7434
5633


KeySpace 100000
Throughput Time (in MilliSeconds)
No of threads
Distribution A
Distribution B
Distribution C
42105
31067
33542
22008
28828
24316
23323
17241
11173
12924
6633
5128
5649
3638
5174
4395



Observations:

From the above readings, we confirm that:

1. Read dominated operation gives better throughput than either mixed or write dominated operation as it involves lesser locks than the other two types.
2. In Fine Grained Locking, as the no of threads increases, the throughput increases. This is more evident than in coarse grained locking where the throughput does not change significantly as the no of threads increases. This is because the thread locks the BST in coarse grained locking even if no other thread is updating the same region.
3. Also as the key space increases, the throughput decreases in both types of locking.
4. Throughput performance of the fine grained locking on average is ten times better than that of the coarse grained locking.