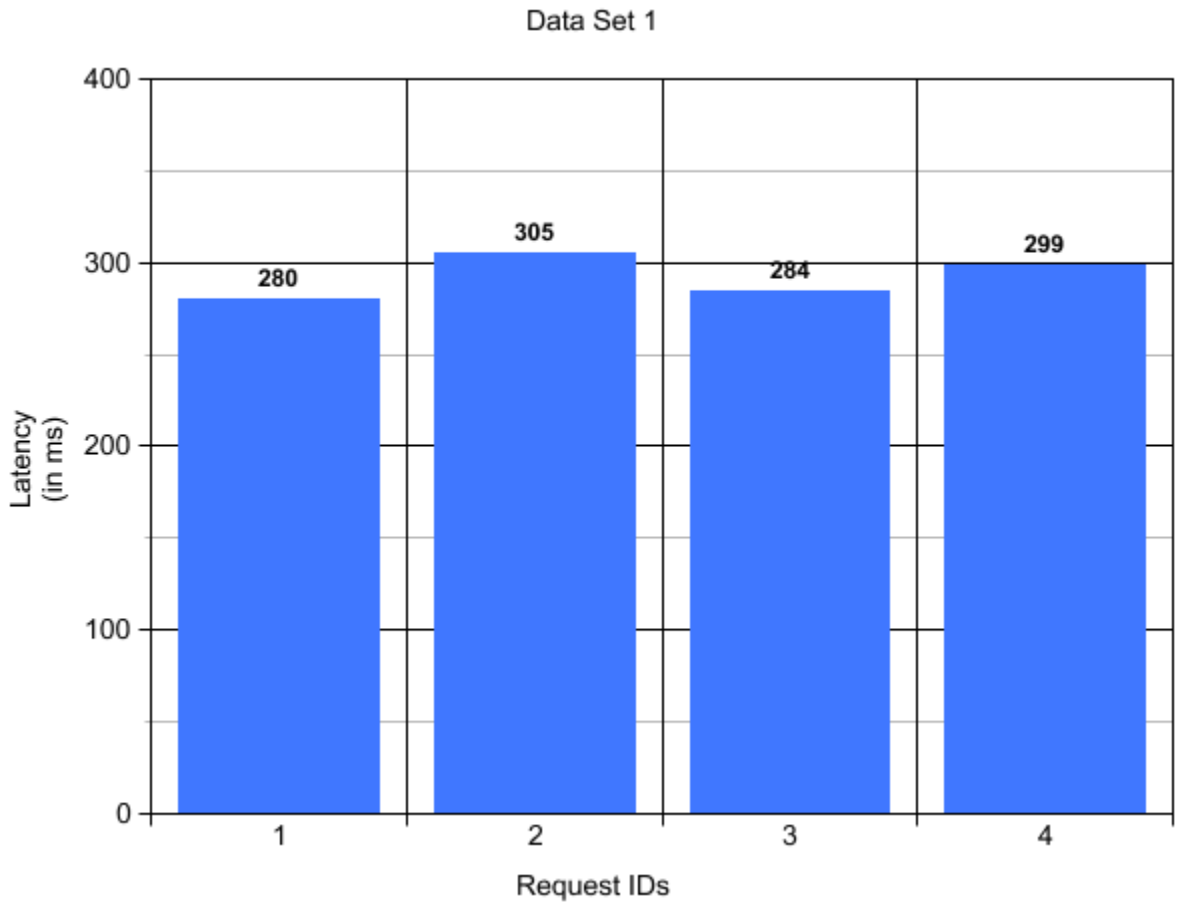
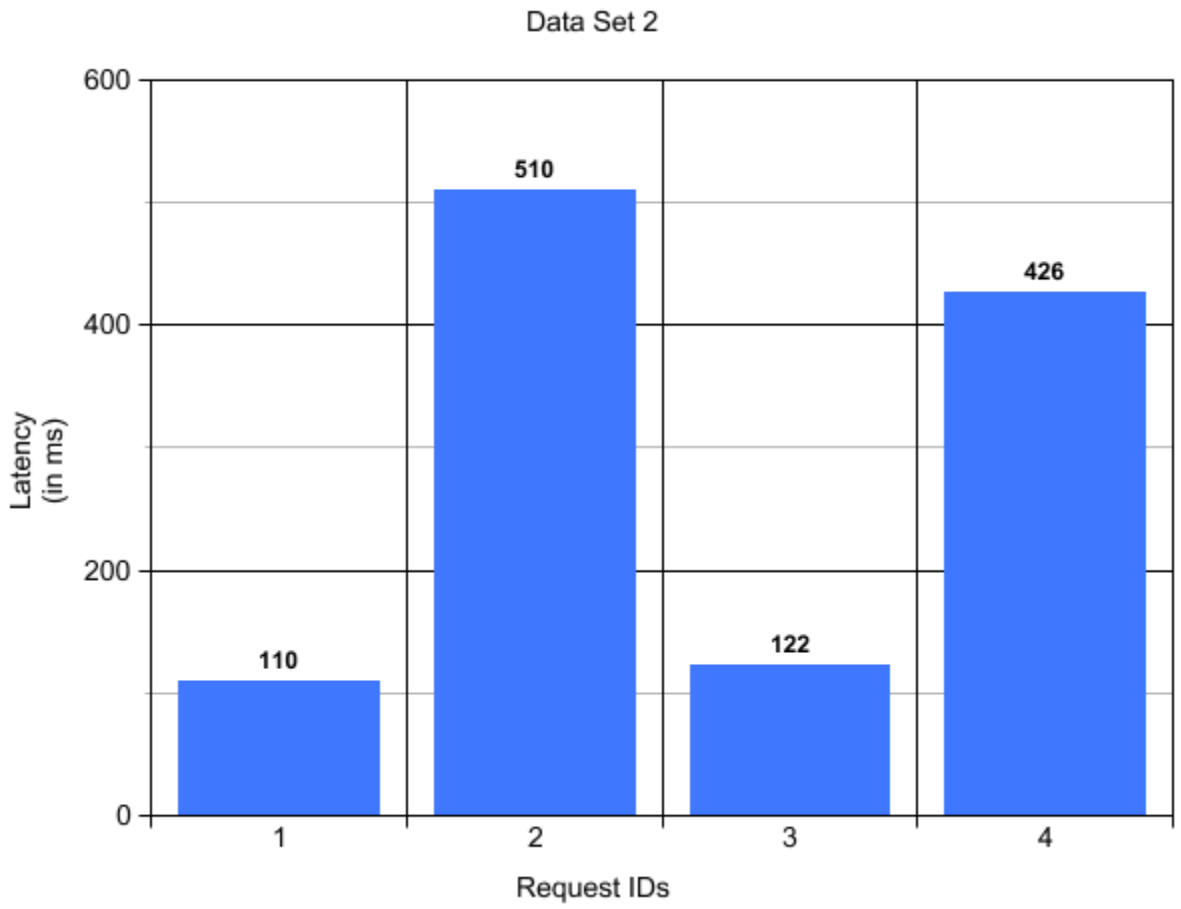
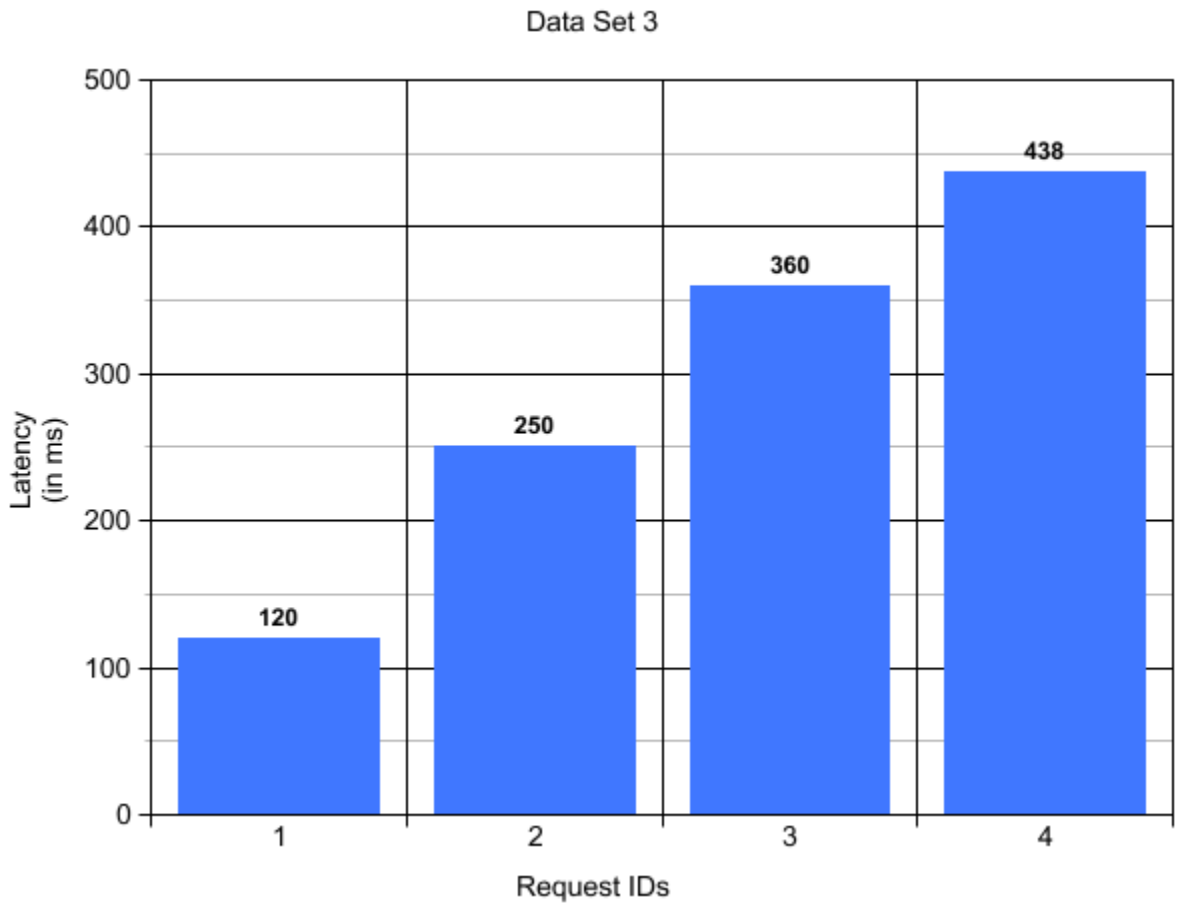
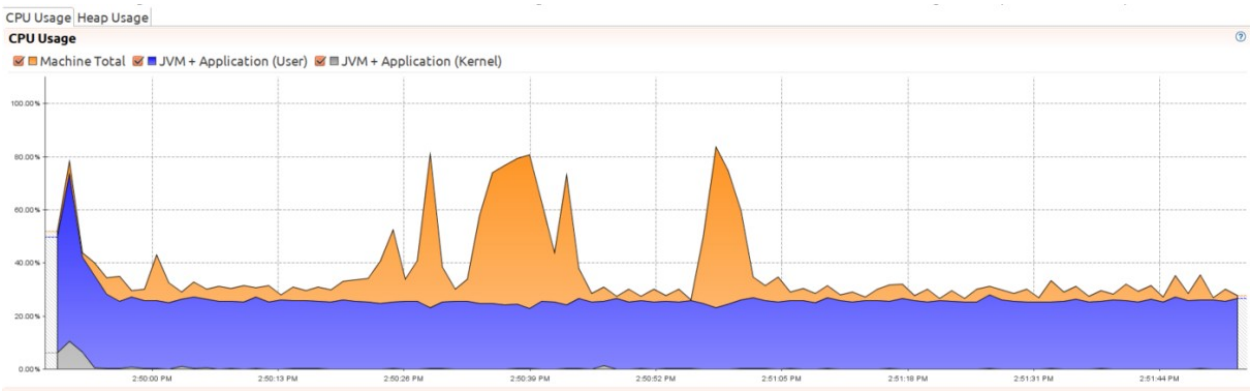
Chapter 1: The Road to Performance

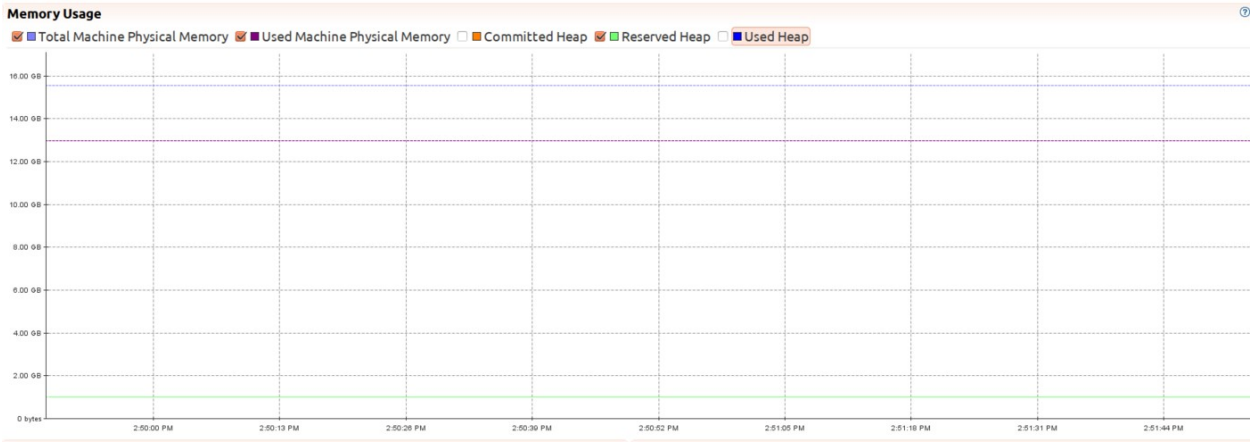


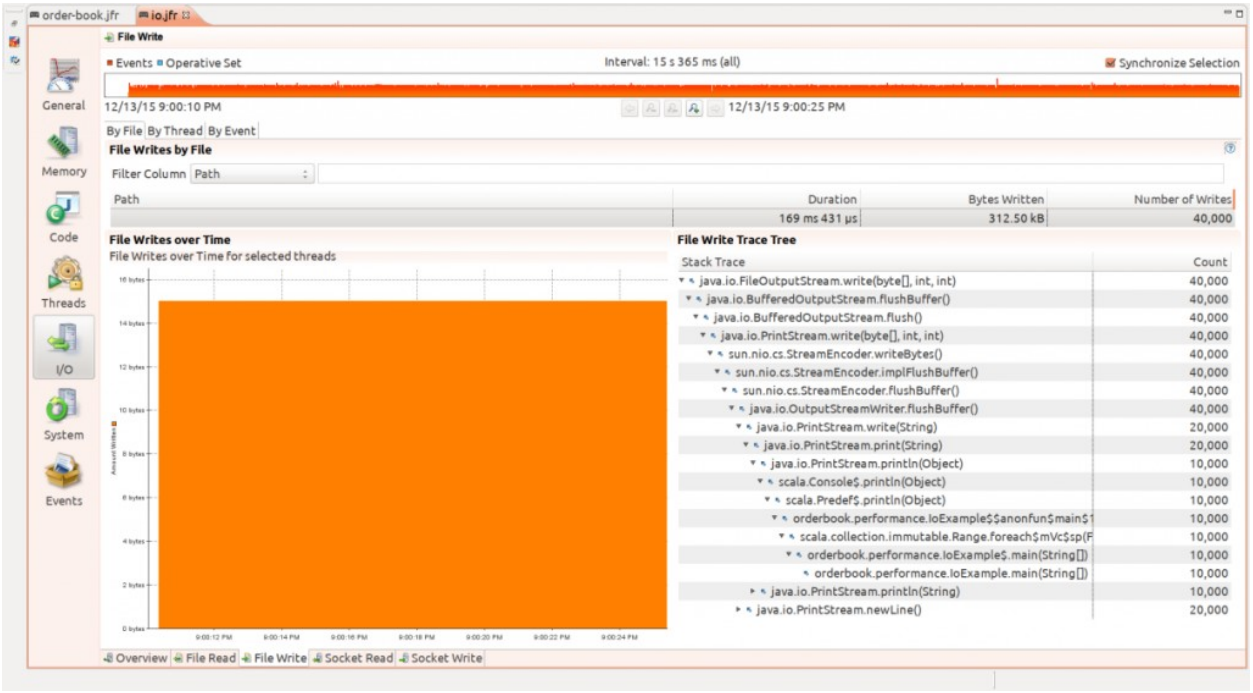

Data Set 2
600
400
200
0
Latency (in ms)
510
426
110
122
1
2
3
4
Request IDs

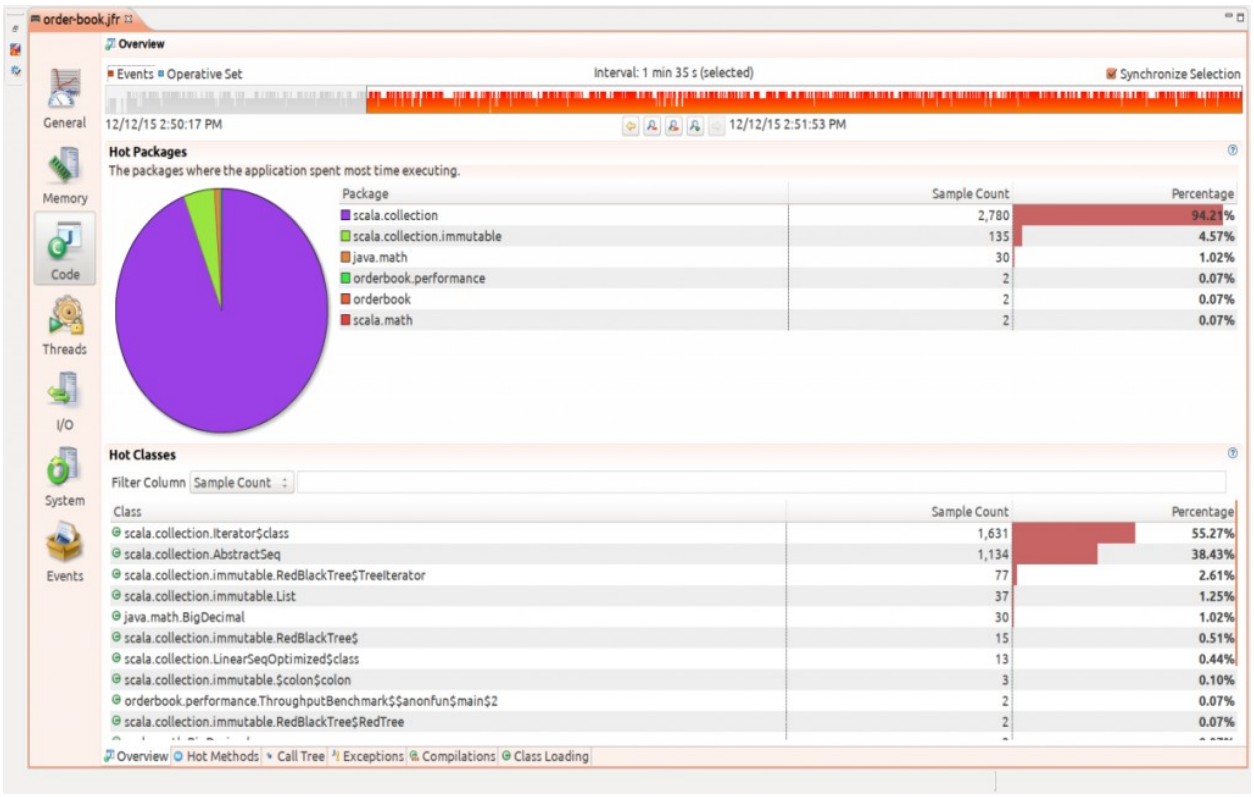


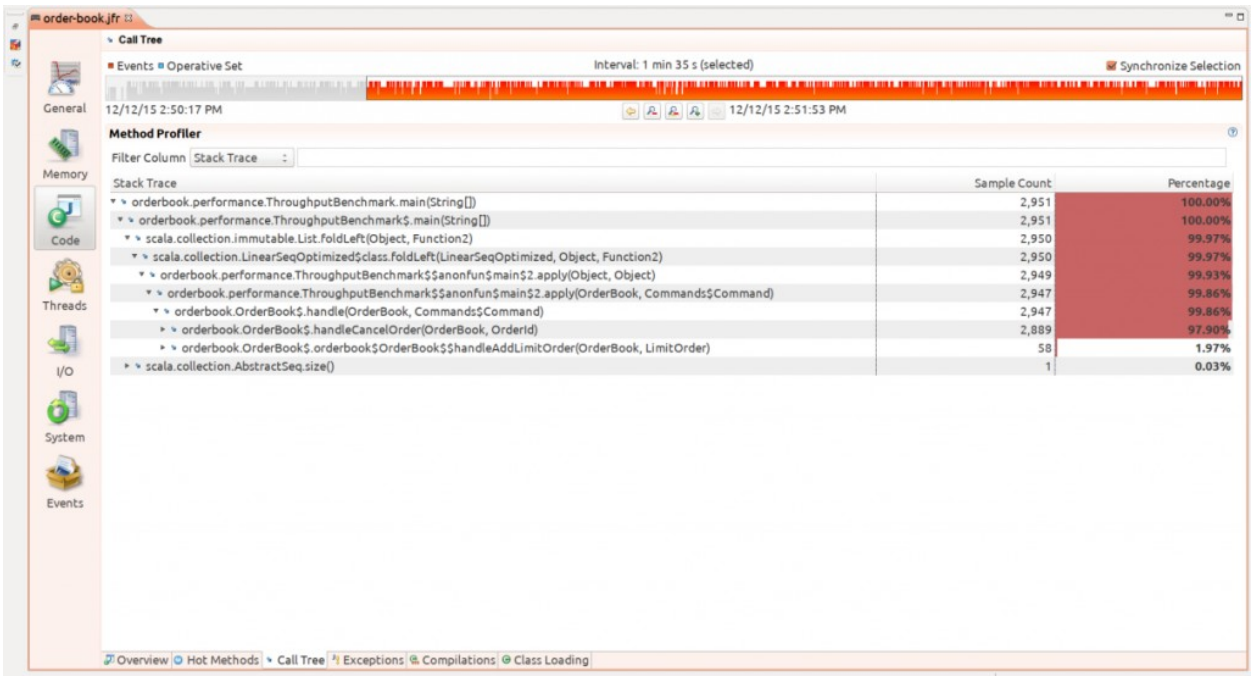
Chapter 2: Measuring Performance on the JVM

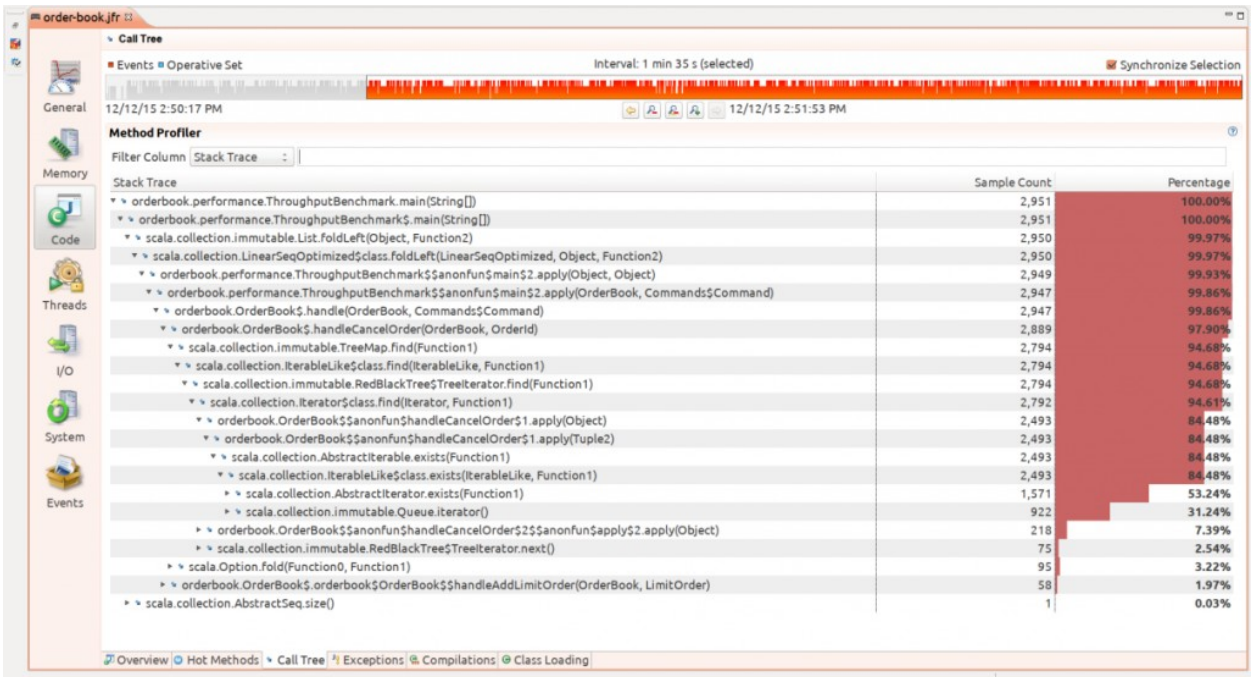


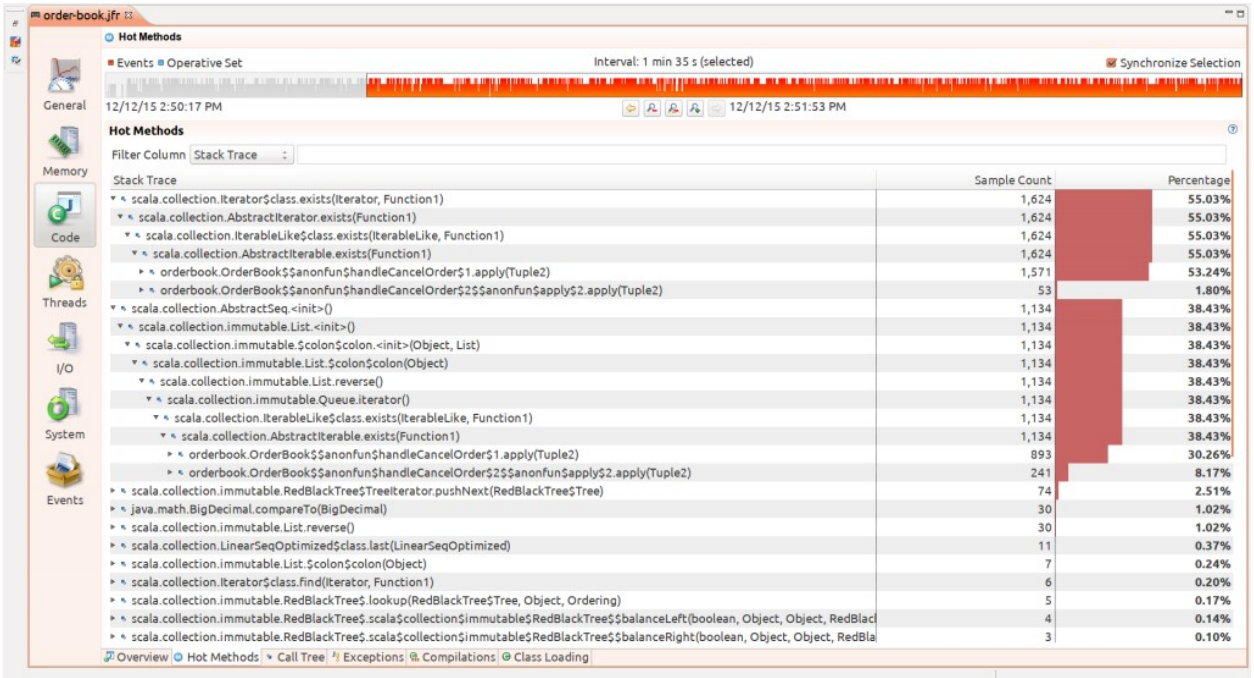

Memory Usage
Total Machine Physical Memory
Used Machine Physical Memory
Committed Heap
Reserved Heap
Used Heap
16.00 GB
14.00 GB
12.00 GB
10.00 GB
8.00 GB
6.00 GB
4.00 GB
2.00 GB
0 bytes
2:50:00 PM
2:50:13 PM
2:50:26 PM
2:50:39 PM
2:50:52 PM
2:51:05 PM
2:51:18 PM
2:51:31 PM
2:51:44 PM

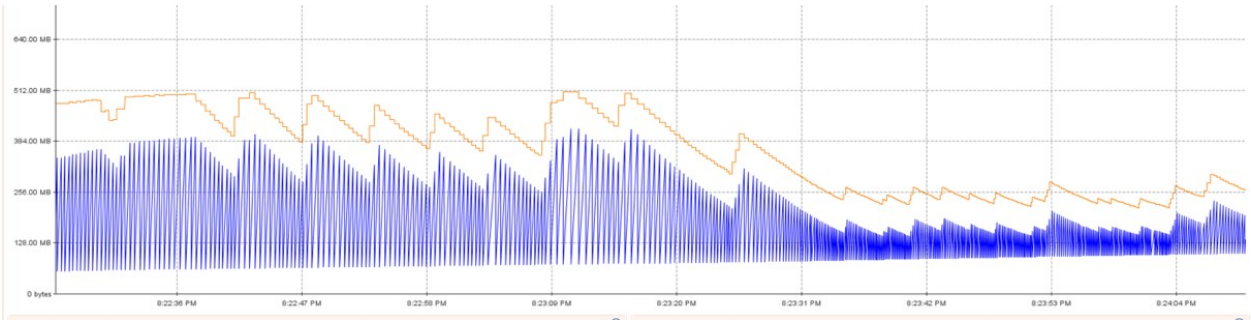

order-book.jfr
io.jfr
File Write
Events
Operative Set
Interval: 15 s 365 ms (all)
Synchronize Selection
General
Memory
Code
Threads
I/O
System
Events
12/13/15 9:00:10 PM
12/13/15 9:00:25 PM
By File
By Thread
By Event
File Writes by File
Filter Column
Path
Path
Duration
Bytes Written
Number of Writes
169 ms 431 µs
312.50 kB
40,000
File Writes over Time
File Writes over Time for selected threads
File Write Trace Tree
Stack Trace
Count
java.io.FileOutputStream.write(byte[], int, int) 40,000
java.io.BufferedOutputStream.flushBuffer() 40,000
java.io.BufferedOutputStream.flush() 40,000
java.io.PrintStream.write(byte[], int, int) 40,000
sun.nio.cs.StreamEncoder.writeBytes() 40,000
sun.nio.cs.StreamEncoder.implFlushBuffer() 40,000
sun.nio.cs.StreamEncoder.flushBuffer() 40,000
java.io.OutputStreamWriter.flushBuffer() 40,000
java.io.PrintStream.write(String) 20,000
java.io.PrintStream.print(String) 20,000
java.io.PrintStream.println(Object) 10,000
scala.Console$.println(Object) 10,000
scala.Predef$.println(Object) 10,000
orderbook.performance.IoExample$$anonfun$main$1 10,000
scala.collection.immutable.Range.foreachmVcsp(F 10,000
orderbook.performance.IoExample$.main(String[]) 10,000
orderbook.performance.IoExample.main(String[]) 10,000
java.io.PrintStream.println(String) 10,000
java.io.PrintStream.newLine() 20,000
Overview
File Read
File Write
Socket Read
Socket Write


order-book.jfr
Overview
General
Memory
Code
Threads
I/O
System
Events
Events
Operative Set
Interval: 1 min 35 s (selected)
Synchronize Selection
12/12/15 2:50:17 PM
12/12/15 2:51:53 PM
Hot Packages
The packages where the application spent most time executing.
Package
Sample Count
Percentage
scala.collection
2,780
94.21%
scala.collection.immutable
135
4.57%
java.math
30
1.02%
orderbook.performance
2
0.07%
orderbook
2
0.07%
scala.math
2
0.07%
Hot Classes
Filter Column
Sample Count
Class
Sample Count
Percentage
scala.collection.Iterator$class
1,631
55.27%
scala.collection.AbstractSeq
1,134
38.43%
scala.collection.immutable.RedBlackTree$TreeIterator
77
2.61%
scala.collection.immutable.List
37
1.25%
java.math.BigDecimal
30
1.02%
scala.collection.immutable.RedBlackTree$
15
0.51%
scala.collection.LinearSeqOptimized$class
13
0.44%
scala.collection.immutable.$colon$colon
3
0.10%
orderbook.performance.ThroughputBenchmark$$anonfun$main$2
2
0.07%
scala.collection.immutable.RedBlackTree$RedTree
2
0.07%
Overview
Hot Methods
Call Tree
Exceptions
Compilations
Class Loading


order-book.jfr
Call Tree
General
Memory
Code
Threads
I/O
System
Events
Events
Operative Set
Interval: 1 min 35 s (selected)
Synchronize Selection
12/12/15 2:50:17 PM
12/12/15 2:51:53 PM
Method Profiler
Filter Column
Stack Trace
Stack Trace
Sample Count
Percentage
orderbook.performance.ThroughputBenchmark.main(String[])
2,951
100.00%
orderbook.performance.ThroughputBenchmark$.main(String[])
2,951
100.00%
scala.collection.immutable.List.foldLeft(Object, Function2)
2,950
99.97%
scala.collection.LinearSeqOptimized$class.foldLeft(LinearSeqOptimized, Object, Function2)
2,950
99.97%
orderbook.performance.ThroughputBenchmark$$anonfun$main$2.apply(Object, Object)
2,949
99.93%
orderbook.performance.ThroughputBenchmark$$anonfun$main$2.apply(OrderBook, Commands$Command)
2,947
99.86%
orderbook.OrderBook$.handle(OrderBook, Commands$Command)
2,947
99.86%
orderbook.OrderBook$.handleCancelOrder(OrderBook, OrderId)
2,889
97.90%
orderbook.OrderBook$.orderbook$OrderBook$$handleAddLimitOrder(OrderBook, LimitOrder)
58
1.97%
scala.collection.AbstractSeq.size()
1
0.03%
Overview
Hot Methods
Call Tree
Exceptions
Compilations
Class Loading


order-book.jfr
Call Tree
Events
Operative Set
Interval: 1 min 35 s (selected)
Synchronize Selection
12/12/15 2:50:17 PM
12/12/15 2:51:53 PM
General
Memory
Code
Threads
I/O
System
Events
Method Profiler
Filter Column
Stack Trace
Sample Count
Percentage
orderbook.performance.ThroughputBenchmark.main(String[]) 2,951 100.00%
orderbook.performance.ThroughputBenchmark$.main(String[]) 2,951 100.00%
scala.collection.immutable.List.foldLeft(Object, Function2) 2,950 99.97%
scala.collection.LinearSeqOptimized$class.foldLeft(LinearSeqOptimized, Object, Function2) 2,950 99.97%
orderbook.performance.ThroughputBenchmark$$anonfun$main$2.apply(Object, Object) 2,949 99.93%
orderbook.performance.ThroughputBenchmark$$anonfun$main$2.apply(OrderBook, Commands$Command) 2,947 99.86%
orderbook.OrderBook$.handle(OrderBook, Commands$Command) 2,947 99.86%
orderbook.OrderBook$.handleCancelOrder(OrderBook, OrderId) 2,889 97.90%
scala.collection.immutable.TreeMap.find(Function1) 2,794 94.68%
scala.collection.IterableLike$class.find(IterableLike, Function1) 2,794 94.68%
scala.collection.immutable.RedBlackTree$TreeIterator.find(Function1) 2,794 94.68%
scala.collection.Iterator$class.find(Iterator, Function1) 2,792 94.61%
orderbook.OrderBook$$anonfun$handleCancelOrder$1.apply(Object) 2,493 84.48%
orderbook.OrderBook$$anonfun$handleCancelOrder$1.apply(Tuple2) 2,493 84.48%
scala.collection.AbstractIterable.exists(Function1) 2,493 84.48%
scala.collection.IterableLike$class.exists(IterableLike, Function1) 2,493 84.48%
scala.collection.AbstractIterator.exists(Function1) 1,571 53.24%
scala.collection.immutable.Queue.iterator() 922 31.24%
orderbook.OrderBook$$anonfun$handleCancelOrder$2$$anonfun$apply$2.apply(Object) 218 7.39%
scala.collection.immutable.RedBlackTree$TreeIterator.next() 75 2.54%
scala.Option.fold(Function0, Function1) 95 3.22%
orderbook.OrderBook$.orderbook$OrderBook$$handleAddLimitOrder(OrderBook, LimitOrder) 58 1.97%
scala.collection.AbstractSeq.size() 1 0.03%
Overview
Hot Methods
Call Tree
Exceptions
Compilations
Class Loading


order-book.jfr
Hot Methods
Events
Operative Set
Interval: 1 min 35 s (selected)
Synchronize Selection
12/12/15 2:50:17 PM
12/12/15 2:51:53 PM
General
Memory
Code
Threads
I/O
System
Events
Hot Methods
Filter Column
Stack Trace
Sample Count
Percentage
scala.collection.Iterator$class.exists(Iterator, Function1) 1,624 55.03%
scala.collection.AbstractIterator.exists(Function1) 1,624 55.03%
scala.collection.IterableLike$class.exists(IterableLike, Function1) 1,624 55.03%
scala.collection.AbstractIterable.exists(Function1) 1,624 55.03%
orderbook.OrderBook$$anonfun$handleCancelOrder$1.apply(Tuple2) 1,571 53.24%
orderbook.OrderBook$$anonfun$handleCancelOrder$2$$anonfun$apply$2.apply(Tuple2) 53 1.80%
scala.collection.AbstractSeq.<init>() 1,134 38.43%
scala.collection.immutable.List.<init>() 1,134 38.43%
scala.collection.immutable.$colon$colon.<init>(Object, List) 1,134 38.43%
scala.collection.immutable.List.$colon$colon(Object) 1,134 38.43%
scala.collection.immutable.List.reverse() 1,134 38.43%
scala.collection.immutable.Queue.iterator() 1,134 38.43%
scala.collection.IterableLike$class.exists(IterableLike, Function1) 1,134 38.43%
scala.collection.AbstractIterable.exists(Function1) 1,134 38.43%
orderbook.OrderBook$$anonfun$handleCancelOrder$1.apply(Tuple2) 893 30.26%
orderbook.OrderBook$$anonfun$handleCancelOrder$2$$anonfun$apply$2.apply(Tuple2) 241 8.17%
scala.collection.immutable.RedBlackTree$TreeIterator.pushNext(RedBlackTree$Tree) 74 2.51%
java.math.BigDecimal.compareTo(BigDecimal) 30 1.02%
scala.collection.immutable.List.reverse() 30 1.02%
scala.collection.LinearSeqOptimized$class.last(LinearSeqOptimized) 11 0.37%
scala.collection.immutable.List.$colon$colon(Object) 7 0.24%
scala.collection.Iterator$class.find(Iterator, Function1) 6 0.20%
scala.collection.immutable.RedBlackTree$.lookup(RedBlackTree$Tree, Object, Ordering) 5 0.17%
scala.collection.immutable.RedBlackTree$.scala$collection$immutable$RedBlackTree$$balanceLeft(boolean, Object, Object, RedBlacl 4 0.14%
scala.collection.immutable.RedBlackTree$.scala$collection$immutable$RedBlackTree$$balanceRight(boolean, Object, Object, RedBla 3 0.10%
Overview
Hot Methods
Call Tree
Exceptions
Compilations
Class Loading



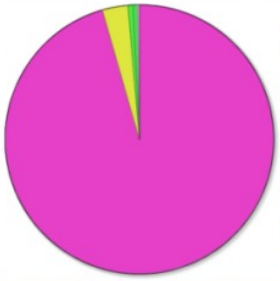
Young Collection Total Time	
GC Count	420
Average GC Time	2 ms 140 µs
Maximum GC Time	17 ms 461 µs
Total GC Time	898 ms 868 µs

Old Collection Total Time	
GC Count	0
Average GC Time	N/A
Maximum GC Time	N/A
Total GC Time	N/A

All Collections Total Time	
GC Count	420
Average GC Time	2 ms 140 µs
Maximum GC Time	17 ms 461 µs
Total GC Time	898 ms 868 µs

Allocation by Class | Allocation by Thread | Allocation Profile

Allocation Pressure



Class	Average Object Size	TLABs	Total TLAB Size	Pressure
scala.collection.immutable.$colon$colon	24 bytes	20,520	57.34 GB	95.59%
scala.Tuple2	24 bytes	536	1.80 GB	3.00%
scala.collection.immutable.RedBlackTree$BlackTree	32 bytes	127	379.67 MB	0.62%
scala.collection.immutable.RedBlackTree$RedTree	32 bytes	72	254.39 MB	0.41%
scala.Some	16 bytes	11	45.54 MB	0.07%
scala.collection.immutable.TreeMap	24 bytes	13	43.07 MB	0.07%
scala.collection.immutable.Queue$$anonfun$newBuilder$1	16 bytes	7	25.95 MB	0.04%
scala.collection.immutable.RedBlackTree$EntriesIterator	32 bytes	8	25.02 MB	0.04%
scala.collection.immutable.Queue	24 bytes	8	21.41 MB	0.03%
orderbook.OrderBook	24 bytes	8	20.26 MB	0.03%
scala.collection.immutable.RedBlackTree$Tree[]	96 bytes	7	16.92 MB	0.03%
scala.collection.mutable.Builder$$anon$1	24 bytes	5	14.06 MB	0.02%
orderbook.OrderBook$$anonfun$handleCancelOrder$3	24 bytes	3	10.89 MB	0.02%

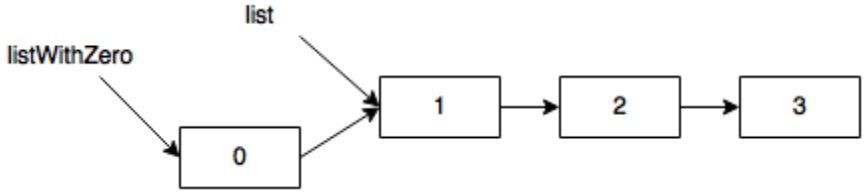
Stack Trace

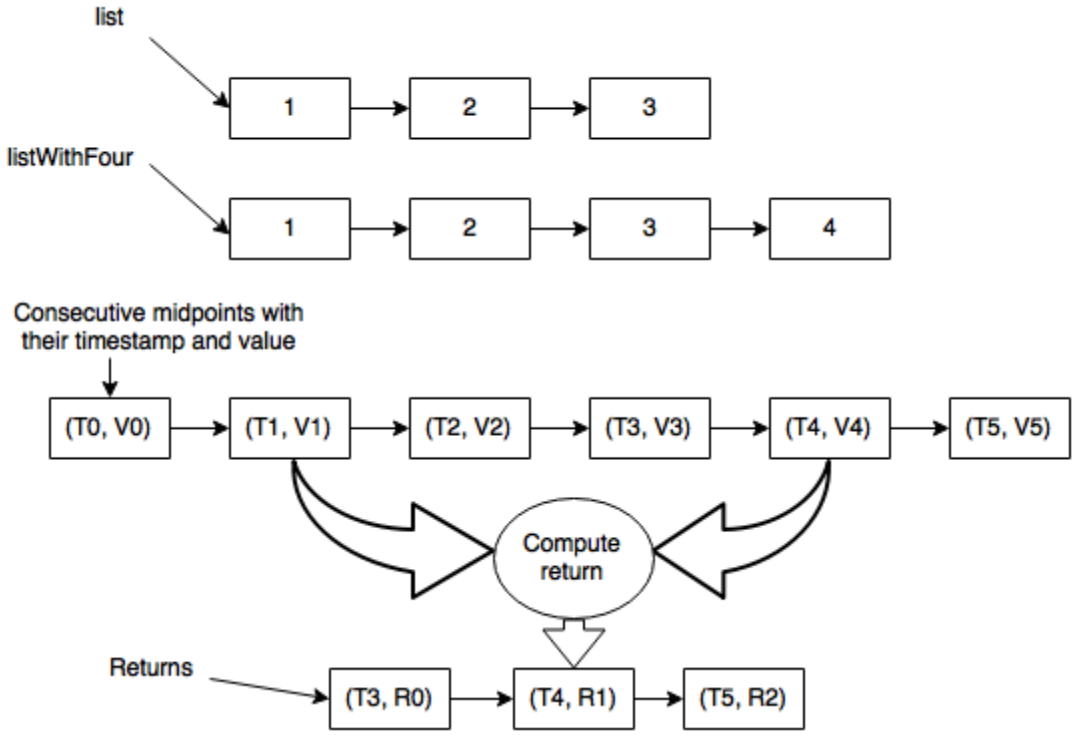
Stack Trace	TLABs	Total TLAB Size	Pressure
scala.collection.immutable.List.$colon$colon(Object)	20,520	57.34 GB	100.00%
scala.collection.immutable.List.reverse()	20,516	57.34 GB	99.99%
scala.collection.immutable.Queue.iterator()	20,516	57.34 GB	99.99%
scala.collection.IterableLike$class.exists(IterableLike, Function1)	20,514	57.33 GB	99.98%
scala.collection.AbstractIterable.exists(Function1)	20,514	57.33 GB	99.98%
orderbook.OrderBook$$anonfun$handleCancelOrder$1.apply(Tuple2)	20,112	56.24 GB	98.08%
orderbook.OrderBook$$anonfun$handleCancelOrder$2$$anonfun$apply$2.apply(Tuple2)	402	1.09 GB	1.90%
scala.collection.IterableLike$class.foreach(IterableLike, Function1)	2	6.68 MB	0.01%
scala.collection.immutable.Queue.enqueue(Object)	4	6.33 MB	0.01%

Allocation Profile

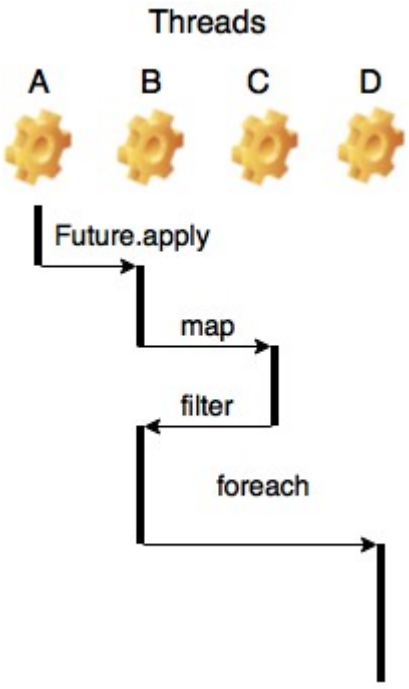
Stack Trace	Average Object Size	Total TLAB size	Pressure
orderbook.performance.ThroughputBenchmark.main(String[])	24 bytes	59.99 GB	100.00%
orderbook.performance.ThroughputBenchmark$.main(String[])	24 bytes	59.99 GB	100.00%
scala.collection.immutable.List.foldLeft(Object, Function2)	24 bytes	59.99 GB	100.00%
scala.collection.LinearSeqOptimized$class.foldLeft(LinearSeqOptimized, Object, Function2)	24 bytes	59.99 GB	100.00%
orderbook.performance.ThroughputBenchmark$$anonfun$2.apply(Object, Object)	24 bytes	59.99 GB	100.00%
orderbook.performance.ThroughputBenchmark$$anonfun$2.apply(OrderBook, Commands$Command)	24 bytes	59.99 GB	100.00%
orderbook.OrderBook$.handle(OrderBook, Commands$Command)	24 bytes	59.99 GB	100.00%
orderbook.OrderBook$.orderbook$OrderBook$$handleAddLimitOrder(OrderBook, LimitOrder)	29 bytes	534.01 MB	0.87%
orderbook.OrderBook$.handleCancelOrder(OrderBook, OrderId)	24 bytes	59.47 GB	99.13%

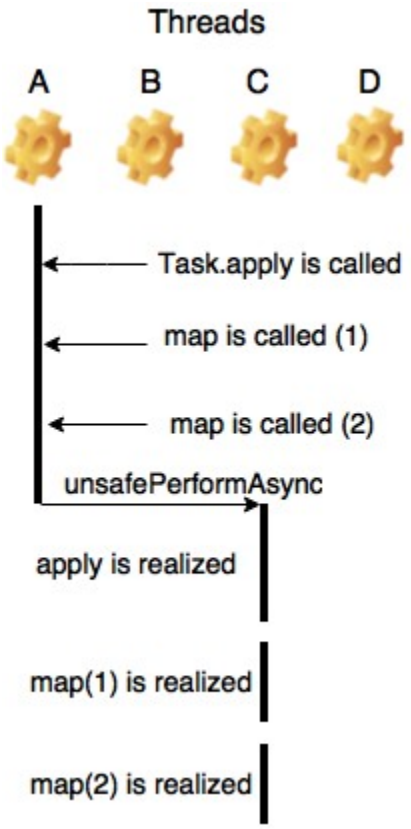
Chapter 4: Exploring the Collection API



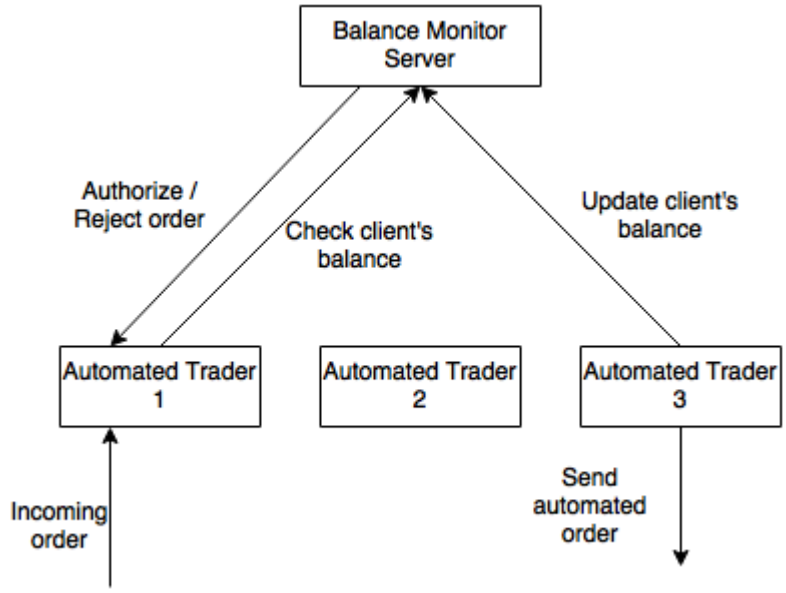


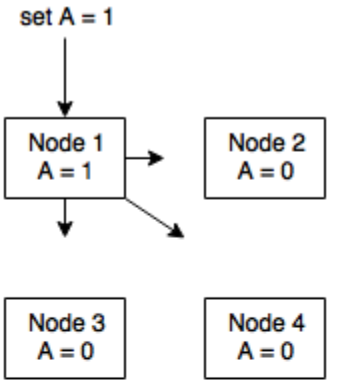
Chapter 6: Concurrency in Scala



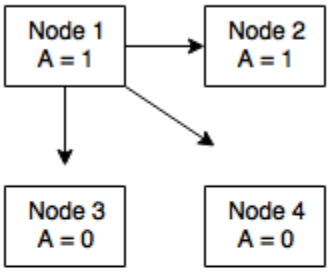


Chapter 7: Architecting for Performance

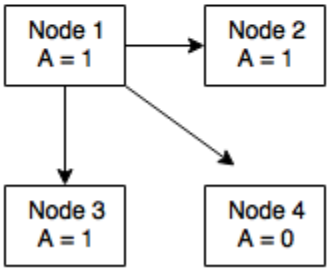




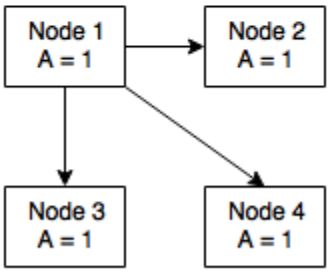
Step 1



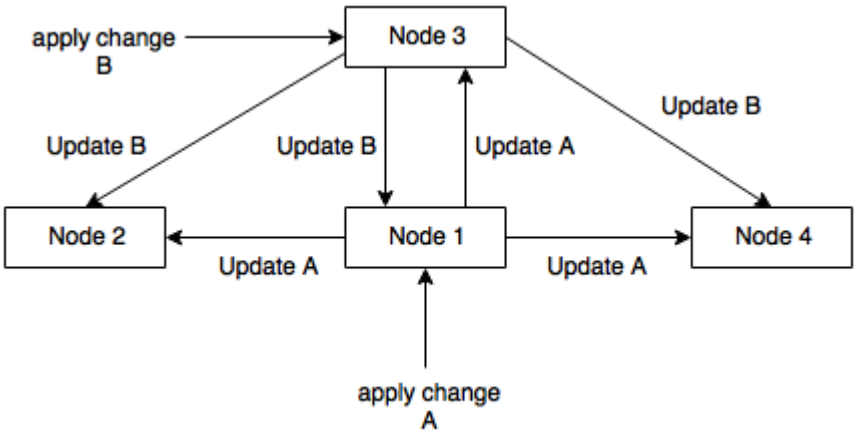
Step 2

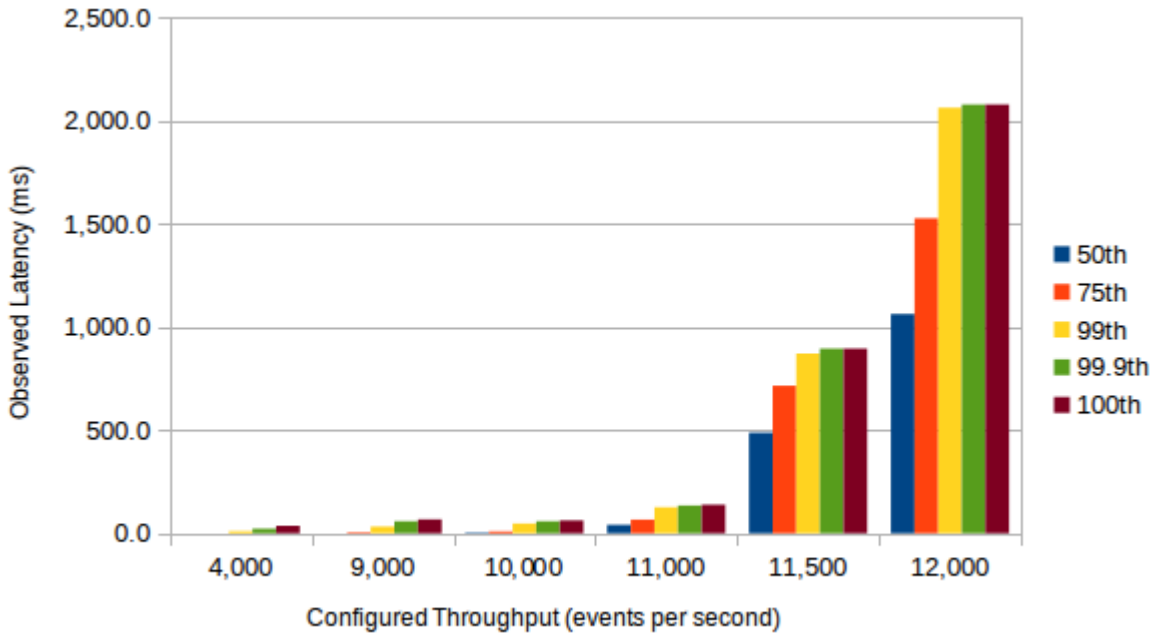


Step 3



Step 4




2,500.0
2,000.0
1,500.0
1,000.0
500.0
0.0
Observed Latency (ms)
4,000
9,000
10,000
11,000
11,500
12,000
Configured Throughput (events per second)
50th
75th
99th
99.9th
100th