The following is an ***excerpt*** from a technical assessment commission by an investment bank to help resolve performance and stability issues experienced during the development and testing of their Coherence cache application but also to provide best practice recommendation for their cache infrastructure.
All relevant details have been anonymised

……previous sections of report omitted

End-of-Day Position Processing and Cached Failure

EOD loading requires a full calculation of the EOD position as illustrated in the diagram below. Up to date real-time position calculation is a principal requirement. In general, synchronous aggregation of associated Position delta (step 2) and intraday (step 3) entries based on PositionKey data affinity is an efficient and dead-lock free operation as long as the backing map is accessed directly by the cache service worker thread, as opposed to a re-entrant call on the cache service, which has deadlock risk.

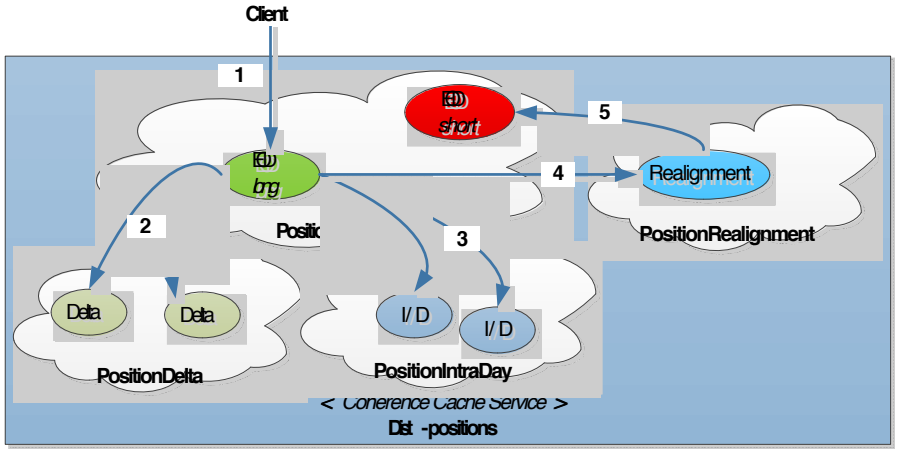

Figure 4 End of day position calculation process

The complication is in the update of realignment quantity available for long (positive) positions (step 4) and a subsequent update of short positions (step 5) with the same ISIN and client hierarchy. Data affinity between realignment and EOD positions is not possible as the realignment key is a subset of the PositionKey and therefore accessing the backing map directly is not possible. As can be seen from the diagram above the realignment cache was also deployed to the dis-positions service. Given the circular nature of EOD calculation processing within the dist-position service, it was suspected that realignment updates where the possible culprit for cache failure.

The cache logging level was increased to confirm the hypothesis and identify the cause of the failure. The excerpts from the cache server log just before and after the cache service thread pool locked up confirm that it is indeed EOD calculation and the realignment update in particular the appears to cause the failure.

```
INFO   | jvm 1    | 2011/10/11 17:50:39 | Oct 11, 2011 4:50:38 PM xxx.xxxxx.xxxxx.cache.calculation.EODCalculator calculate
INFO   | jvm 1    | 2011/10/11 17:50:39 | FINE: PositionKey: PositionKey [ACCOUNT_ID =>JPM0123/789/0098
….rest truncated
```

Approximately 2 minutes later cache service deadlock is detected.

```
INFO   | jvm 1    | 2011/10/11 17:52:27 | Oct 11, 2011 4:52:26 PM xxx.xxxxx.xxxxx.cache.calculation.RealignmentUpdateProcessor process
INFO   | jvm 1    | 2011/10/11 17:52:27 | FINE: No existing realignment. Adding.
INFO   | jvm 1    | 2011/10/11 17:52:27 | 2011-10-11 16:52:26.904/568.391 Oracle Coherence GE 3.6.1.0 <D5>
(thread=Proxy:ExtendTcpProxyService:TcpAcceptorWorker:12, member=1): An exception occurred while processing a InvokeFilterRequest
for Service=Proxy:ExtendTcpProxyService:TcpAcceptor: (Wrapped: Failed request execution for PositionsDistService service on Member)
java.lang.InterruptedException
INFO   | jvm 1    | 2011/10/11 17:52:27 |   at com.tangosol.util.Base.ensureRuntimeException(Base.java:293)
INFO   | jvm 1    | 2011/10/11 17:52:27 |   a                                                                              t
com.tangosol.coherence.component.util.daemon.queueProcessor.service.Grid.tagException(Grid.CDB:36)
INFO   | jvm 1    | 2011/10/11 17:52:27 |   a                                                                              t
com.tangosol.coherence.component.util.daemon.queueProcessor.service.grid.partitionedService.PartitionedCache.onInvokeFilterRequest(
PartitionedCache.CDB:230)
….rest truncated
```

Similarly updates from the realignment to short EOD positions fail due to thread starvation.

```
INFO   | jvm 1    | 2011/10/11 17:52:27 | Oct 11, 2011 4:52:27 PM xxx.xxxxx.xxxxx.cache.calculation.RealignmentUpdateProcessor process
INFO   | jvm 1    | 2011/10/11 17:52:27 | FINE: UPDATE OBJECT: Realignment Key: PositionKey [ACCOUNT_ID =>JPM/CAN/EQU]
```

```
INFO | jvm 1 | 2011/10/11 17:52:27 | 2011-10-11 16:52:26.905/568.392 Oracle Coherence GE 3.6.1.0 <Error>
(thread=PositionsDistServiceWorker:7, member=1): This thread was interrupted while waiting for the results of a request:
INFO | jvm 1 | 2011/10/11 17:52:27 | Poll
INFO | jvm 1 | 2011/10/11 17:52:27 | {
INFO | jvm 1 | 2011/10/11 17:52:27 | PollId=8191, active
INFO | jvm 1 | 2011/10/11 17:52:27 | InitTimeMillis=1318351839028
....rest truncated
```

Introduction of the Realignment Service

The PositionRealignment cache was configured within its own cache service (see Table 2 Cache service configuration for details). The stress tests were repeated using 2, 4, 8, 10, 16, 20 and 50 concurrent client thread with no cache failures observed. The thread timeline below illustrates the Realignment and Position cache service worker threads as the number of concurrent clients are ramped up.

Figure 5 Cache threads timeline after the introduction of a separate Realignment cache service

The cache server thread timeline depicts the execution of the realignment and position worker threads.

EoD Loading Performance Metrics

The table below provides EOD load timings after the introduction of the realignment cache service. The speedup percentages quoted indicate the time speedup over the sequentially loading 1000 EOD positions.

Client Threads	Cache Cluster Size (Nodes)					
	1		2		4	
	Time in ms	Speedup %	Time in ms	Speedup %	Time in ms	Speedup %
1	114922	0.00%	79063	45.35%	60750	89.17%
2	63312	81.52%	41609	176.20%	27797	313.43%
4	33594	242.09%	19750	481.88%	18844	509.86%
8	21921	424.26%	18453	522.78%	18641	516.50%
10	23782	383.23%	19234	497.49%	19532	488.38%
20	23766	383.56%	18500	521.20%	18969	505.84%
50	23281	393.63%	19187	498.96%	19032	503.84%

Table 1 Cache server EOD load timings and speed up for 1000 EOD objects[1]

The chart below illustrates the EOD loading and calculation speedup for varying numbers of concurrent clients and cache servers.

[1] Timing captured on a development/pre component test platform with the following hardware: Intel Xeon 2GHz 8 core processor with 3.5 GB physical memory.

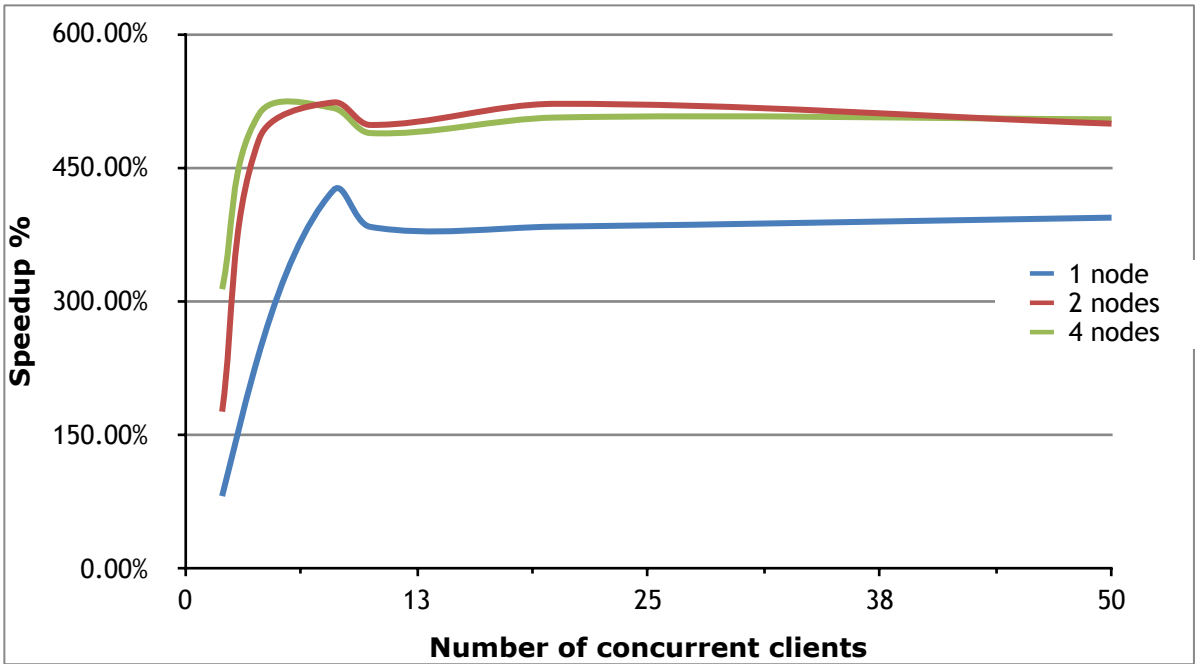

Figure 7 Cache Speedup %

As can be seen from the chart the sweet spot for test platform processor hardware is 8 concurrent client and 2 cache server nodes. This suggests that, all other variables being held constant, an MDB pool size for Position EOD processing should not be greater than the number of raw threads supported by the processor cores within the WebLogic service tier cluster.

……rest of Section omitted

Coherence Cache Configuration and Implementation Considerations

Cache Partitioning
An initial partition size of 257 has been configured. The mathematical theory behind using prime number for the number Map buckets and as part of the hash value generation is long established. The key to hash values is uniqueness and the use of prime values improves the efficacy of your cache and reduces hash collisions.

Best Practice for Cache Keys.
Keys should strictly adhere to the constraints imposed by the java.util.Map interface and ensure that the hashCode() and equals() are consistent. Discontinuities, when the same object attributes are not used in the equals() and hashCode() can lead to a poor performing cache as a result of irregularities in the hashing space.

…..rest of Report omitted