

Performance Tuning and Monitoring for Cognos BI

Cognos BI Performance Team
IBM Cognos Analytics 11.0.13

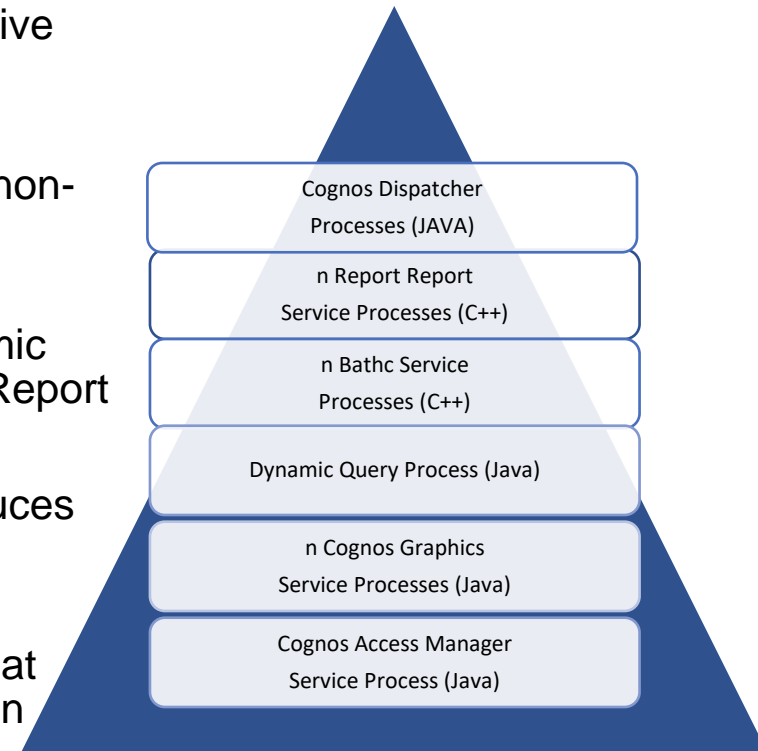
IBM Cognos BI and System Performance

- We have compiled a list of the most common software and hardware bottlenecks and how to monitor the system for them.
 - Process Threading:
 - Too few threads and queuing can occur
 - Process Memory Sizing:
 - Too little memory and OutOfMemory conditions can exist
 - Too little memory and Garbage Collection can become costly
- Nothing in the deck can minimize the impact of a sub-optimal BI model or poorly authored report spec.

IBM Cognos BI Basic Architecture

The following 6 Services account for the core BI processes seen in system process views.

- **Cognos Dispatcher** – Java application responsible for routing requests through the BI system and managing BI content
- **Report Service** – C++ application that manages interactive requests to execute reports (Process name BIBusTKServerMain.exe)
- **Batch Report Service** – C++ application that manages non-interactive requests to execute reports (Process name: BIBusTKServerMain.exe)
- **Dynamic Query** – Java application that manages Dynamic Query requests and returns the result to the requesting Report Service or Batch Report Service
- **Cognos Graphics Service** – Java application that produces graphics on behalf of the Report service (Process name: Windows - cgsLauncher.exe, *nix - Java)
- **Cognos Access Manager Service** – Java application that handles user authentication, authorization, and encryption (Process name: CAM_LPSvr)



BI Report Service Tuning

- Key things to consider when tuning Report Service:
 - If Report Service processes are not enough for the load we are running with on the system queuing can occur.
 - Queuing of requests negatively impacts the BI system,
 - Would affect both Interactive Report Service and Batch Report Service.
 - Affects Dynamic and Compatible Reporting Engines.
 - Cognos Workspace (parallel report execution) generally utilizes more Report Service connections per user than Cognos Viewer.

Metrics - System

0 0 0 No metric score

Queue - Report service

Latency	00:00:00.015
Number of queue requests	802
Queue length	-- 39
Queue length high watermark	39
Queue length low watermark	1
Time in queue	00:00:12.076
Time in queue high watermark	00:00:48.223

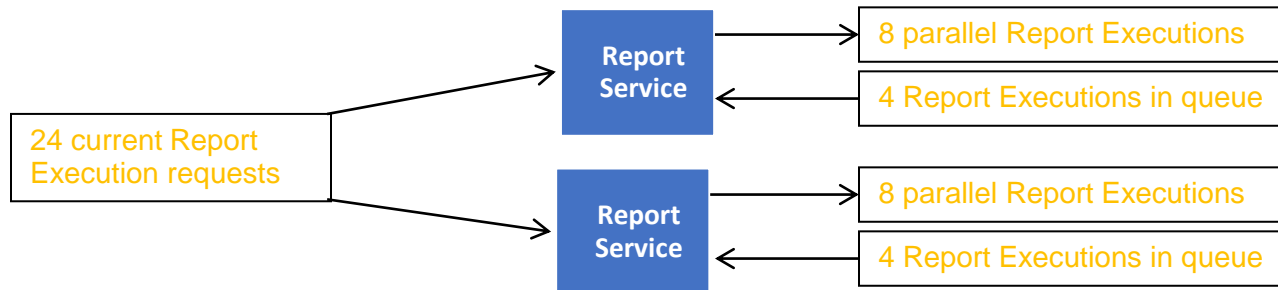
39 users in queue

Waiting 12s

BI Report Service Tuning

Bottlenecks in BI

- The threading model in 10.2.2 allows for 8 low affinity threads and 2 high affinity threads
- 2 Report Server processes map to 16 low affinity threads (Default value).
- If 24 requests are issued to report service, 16 get served and 8 will get queued.



- Increase the number of Report Server processes through Cognos Administration

Set properties - ReportService

General Settings Permissions

Category	Name	Value	Acqu
Tuning	Maximum number of processes for the report service during non-peak period	2	Yes
Tuning	Maximum number of processes for the report service during peak period	2	Yes

Increase the value to 19

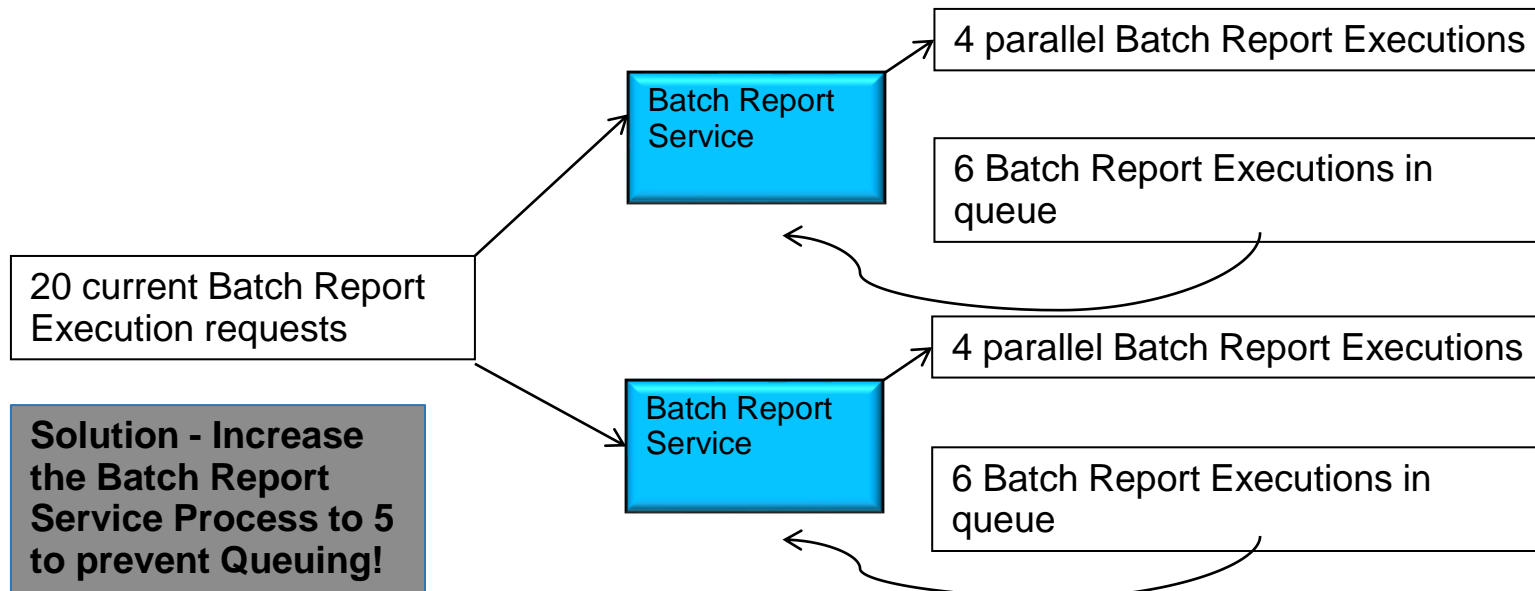
Batch Report Service 10.2.2 Tuning

- Similar to the Report Service. The Batch Report Service handles job report execution. The number of processes has a significant effect on batch report execution
 - Too few Batch Report Service processes will lead to report execution requests waiting in the Queue.
- Set in IBM Cognos Administration:

Set properties - BatchReportService

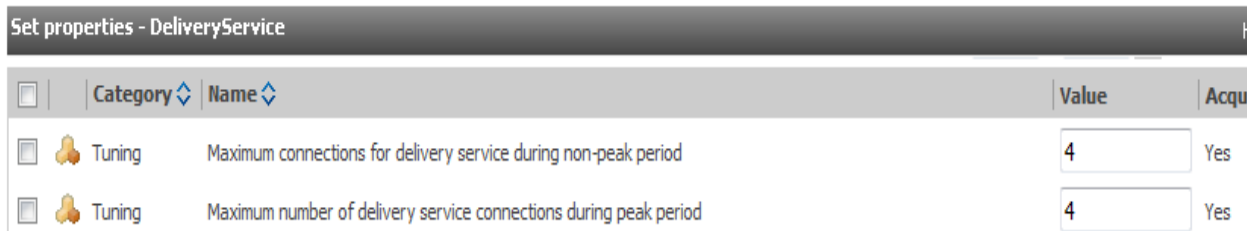
General Settings Permissions

Category	Name	Value	Acqu
Tuning	Maximum number of processes for the batch report service during non-peak period	2	Yes
Tuning	Maximum number of processes for the batch report service during peak period	2	Yes



Batch Report Service 10.2.2 Tuning

- The number of Delivery Service connections may need to be increased for heavy Batch environments that write to disk:



The screenshot shows a dialog box titled "Set properties - DeliveryService". It contains a table with two rows of tuning parameters. Each row has a checkbox, a category icon (a person), the category name "Tuning", a description, a value input field containing "4", and an "Acqu" checkbox.

<input type="checkbox"/>	Category	Name	Value	Acqu
<input type="checkbox"/>	Tuning	Maximum connections for delivery service during non-peak period	4	Yes
<input type="checkbox"/>	Tuning	Maximum number of delivery service connections during peak period	4	Yes

JVM Tuning for BI

- There are 3 main java processes associated with Cognos BI:
 - Dispatcher JVM
 - Dynamic Query JVM
 - Cognos Graphics Service JVM
- There are 2 key performance parameters for the JVM processes:
 - JVM Settings
 - Initial Heap Size (Xms) = Maximum Heap Size (Xmx)
 - Xmn
 - Xgcpolicy:gencon <http://javaeesupportpatterns.blogspot.com/2012/03/ibm-jvm-tuning-gencon-gc-policy.html>
 - Thread Pools (1500)
- Non-optimal settings in these areas can lead to:
 - OutOfMemory conditions
 - Frequent or long Garbage Collection pauses
 - 'timeout' or 'unresponsive' application
 - Overall poor performance and queuing

JAVA Heap
- Xms
- Xmx
- Xmn
-Xgcpolicy:gencon
-Xcompressedrefs

Dispatcher Tuning

JAVA Heap
- Xms
- Xmx
- Xmn
-Xgcpolicy:gencon
-Xcompressedrefs

- **WebSphere Liberty**

- JVM size is configured in Cognos Configuration (1536MB default)
- Gencon and Xcompressedref applied by default in 10.2.2
- Any extra JVM arguments configured in bootstrap_<os>.xml file
- Threading by default is configured for high load testing (1500 threads) in the config tool file

- **WebSphere**

- JVM size and arguments are configured in the WebSphere Admin Web Portal
- Gencon and Xcompressedrefs are not applied by default
- Threading is configured in the WebSphere Admin Portal. Default is 50 threads – too small for high load testing
- Websphere “Allow thread allocation beyond maximum thread size” – use at own risk.
- Ensure the WebSphere version is up to date. Old WebSphere Java versions can significantly affect Cognos BI

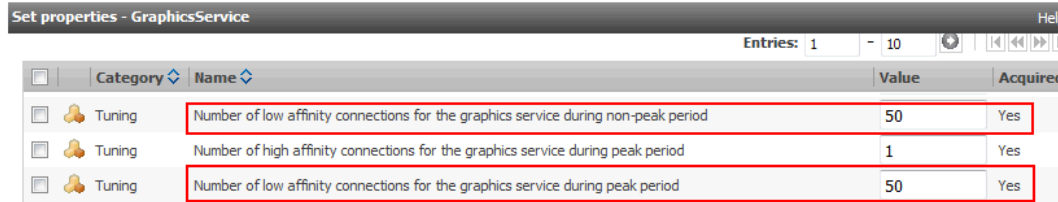
Dynamic Query (DQ) Tuning

- JVM size and JVM arguments are configured in Cognos Administration
- Default JVM size of 1GB
- Xgencon and Xcompressedrefs applied to the JVM by default
- Threading is handled dynamically by the Query Engine. The number of Interactive Report Services and Batch Report Services affect the thread count.
- DQ automatically provides logs (dq_verbosegc_<timestamp>.log) to help determine if your values are set correctly and allow for easy debugging.

JAVA Heap
- Xms
- Xmx
-Xmn
-Xgcpolicy:gencon
-Xcompressedrefs

Cognos Graphics Service (CGS) Tuning

- Threading configured in Cognos Administration. 50 threads per process.



The screenshot shows a table with the following data:

Category	Name	Value	Acquired
Tuning	Number of low affinity connections for the graphics service during non-peak period	50	Yes
Tuning	Number of high affinity connections for the graphics service during peak period	1	Yes
Tuning	Number of low affinity connections for the graphics service during peak period	50	Yes

JAVA Heap
- Xms
- Xmx
- Xmn
-Xgcpolicy:gencon
-Xcompressedrefs

- Default JVM size of 1GB with no JVM arguments applied.
- **Unix / Linux**
 - JVM values and arguments configured in cgsServer.sh in the bin and bin64 locations. JVM arguments set after \$JAVA_OPTS:
- **Windows**
 - JVM values and arguments configured in cgsService.xml in /webapps/p2pd/WEB-INF/services. JVM arguments set between child-proc-cmd tags after vmargs (2 places in the same file):

\$JAVA_OPTS -Xmx4g -Xms4g -Xmn2g -Xcompressedrefs -Xgcpolicy:gencon

```
<child-proc-cmd>vmargs</child-proc-cmd>  
<child-proc-cmd>Xmx2g</child-proc-cmd>  
<child-proc-cmd>Xms2g</child-proc-cmd>  
<child-proc-cmd>Xmn1g</child-proc-cmd>  
<child-proc-cmd>Xcompressedrefs</child-proc-cmd>  
<child-proc-cmd>Xgcpolicy:gencon</child-proc-cmd>
```

Monitoring Tools: GC Logs

JVM Memory and GC Policies

- Enabling GC logging is a low impact method of measuring JVM sizing and Garbage Collection policies. Undersized JVMs can lead to OOM situations or excessive garbage collections and high JVM pause times.
- For Dispatcher and CM, edit bin64\bootstrap_<OS>.xml and add the following line to the Java arg list
- For CGS, edit \webapps\p2pd\WEB-INF\services\cgsService.xml or cgsServer.sh on UNIX and add to the JVM arguments (in two locations in the file!)

```
-Xverbosegclog:../logs/verbosegc_CGS_%Y%m%d.%H%M%S.%pid.log,10,10000
```

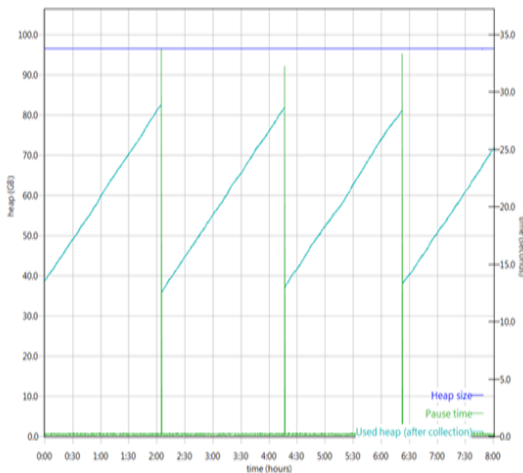
- For Dynamic Query and Dynamic Cubes, GC Logging is on by default in a file named dq_verbosegc_%timeStamp%.log

GC Logging output from IBM Support Assistant

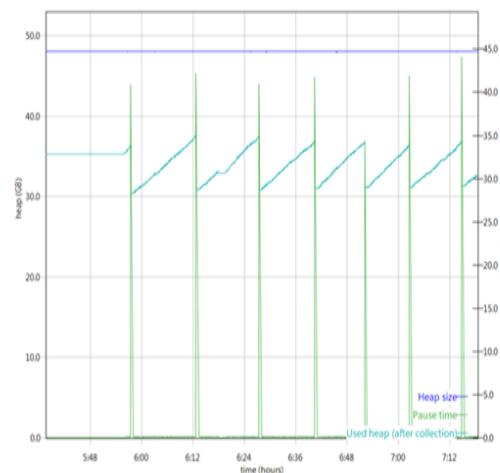
- Classic example of the IBM Support Assistant helping determine 'ideal' JVM size
 - Left graph shows a properly tuned JVM with GC occurring approximately every 2 hours with a pause time of under 35 seconds.
 - Right graph illustrates a JVM running an undersized Java Heap. GC occurs every 10 minutes with a pause time of over 40 seconds.

Healthy BI System

Tuned JVM (using Xms/Xmx = 100 GB)



Small JVM (using Xms/Xmx = 50 GB)



Summary

Concurrent collection count	1
Forced collection count	0
GC Mode	gencon
Global collections - Mean garbage collection pause (ms)	543
Global collections - Mean interval between collections (ms)	1534407
Global collections - Number of collections	1
Global collections - Total amount tenured (MB)	421
Largest memory request (bytes)	6813808
Number of collections triggered by allocation failure	16
Nursery collections - Mean garbage collection pause (ms)	469
Nursery collections - Mean interval between collections (ms)	163418
Nursery collections - Number of collections	16
Nursery collections - Total amount flipped (MB)	1615
Nursery collections - Total amount tenured (MB)	433
Proportion of time spent in garbage collection pauses (%)	0.62
Proportion of time spent unpaused (%)	99.38
Rate of garbage collection (MB/minutes)	127

Operating System Monitoring for BI

- Important to use tools that allow for unattended monitoring of resource utilization over time. For example:
 - PerfMon for Windows
 - Nmon on AIX/Linux (use Nmon Analyser to process Nmon output)
- Read the Cognos BI documentation for any OS specific settings that may need to be applied.
- Bottlenecks in the system resources can lead to frustration in the BI community due to:
 - Inconsistent performance
 - Unexpected error messages
- As hardware and software evolve, system bottlenecks tend to shift. The four most common system bottlenecks are:
 - CPU
 - Memory for both the entire system and key BI processes
 - Network utilization
 - Disk for read, writes, and waits

Operating System Monitoring for BI

- If CPU is a bottleneck:
 - Shift BI services to other servers in the system that may have available CPU
 - Consider adding additional CPU resources or an additional server
 - Monitor Run Queue
- If memory is a bottleneck:
 - Turn off services and processes to free up memory
 - Add more memory to the system or shift BI services to servers with more available RAM
 - Do not rely on Virtual Memory. Absolute performance killer!
- If network appears to be a bottleneck:
 - Check that the NIC card is using the full bandwidth available
 - Ensure the server resolves localhost locally first and not to the DNS first (netshvc.conf)
 - Ensure routers in the BI system are not overtaxed
- If disk might be a bottleneck:
 - Check that file system logging is turned off or minimized (mount)
 - Consider using fast storage to host disk intensive BI services

Appendix 1: IBM Support Assistant

- The gathered Garbage Collection Logs can be viewed using a free tool from IBM called 'IBM Support Assistant Workbench'.
 - Download from <http://www-01.ibm.com/software/support/isa/workbench.html>
 - The desired toolset is 'IBM Monitoring and Diagnostic Tools for Java'

IBM Monitoring and Diagnostic Tools for Java™ - Dump Analyzer

IBM Monitoring and Diagnostic Tools for Java™ - Garbage Collection and Memory Visualizer for ISAv4

IBM Monitoring and Diagnostic Tools for Java™ - Health Center

IBM Monitoring and Diagnostic Tools for Java™ - Interactive Diagnostic Data Explorer

IBM Monitoring and Diagnostic Tools for Java™ - Memory Analyzer

IBM Monitoring and Diagnostic Tools for Java™ - Memory Analyzer 64bit

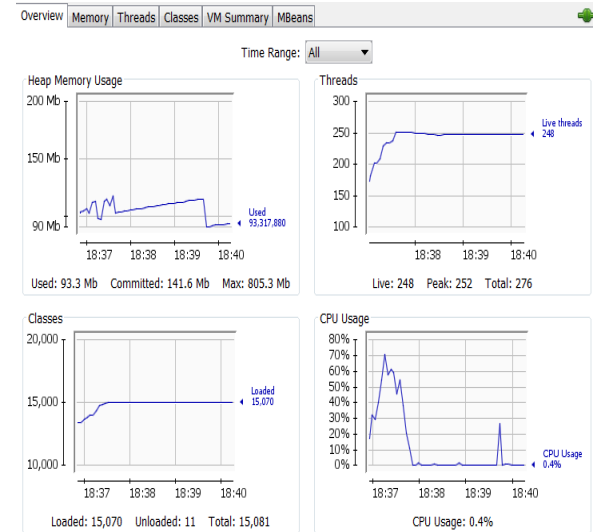
- Quick steps to use the tool:
 - Highlight the 'Garbage Collection and Memory Visualizer' tool and hit 'Launch' and browse to the garbage collection log using the 'Remote Artifact Browser'
 - Arguably the most useful view is 'Heap size', 'Pause time', and 'Used heap (after collection)' chosen from the File menu 'VGC pause' and 'VGC heap'

Appendix 2: Monitoring Tools: JConsole

- Edit `\webapps\p2pd\WEB-INF\services\cgsService.xml` and in two locations add the following args at the end of the `<JVM arguments>` section

```
<child-proc-cmd>Dcom.sun.management.jmxremote</child-proc-cmd>  
<child-proc-cmd>Dcom.sun.management.jmxremote.port=6999</child-proc-cmd>  
<child-proc-cmd>Dcom.sun.management.jmxremote.ssl=false</child-proc-cmd>  
<child-proc-cmd>Dcom.sun.management.jmxremote.authenticate=false</child-proc-cmd>  
<child-proc-cmd>Djava.rmi.server.hostname=tp-pilonmw7</child-proc-cmd>
```

- View the current Java internals using `jconsole hostname:jmxport`
E.G.: `jconsole tp-pilonmw7:6999`



- JConsole is one of ways to get internal metrics from the BI Query Service.

The screenshot shows the IBM Cognos Administration console. The main content area displays the "Set properties - QueryService" configuration page. The "Configuration" tab is selected. The page lists several tuning properties:

Property	Value
Additional JVM arguments for the query service (Requires QueryService restart)	-
Number of garbage collection cycles output to the verbose log (Requires QueryService restart)	1
Disable JVM verbose garbage collection logging (Requires QueryService restart)	<input type="checkbox"/>

IBM Cognos BI Tuning Suggestions

Hardware Configuration:

CPU: 64 cores

RAM: 256 GB

- Information provided.
 - **Peak number of reports/day:** ~8000, majority are interactive reports
 - **Number of active users:** ~1500 that have run at least 1 report in that day
 - **Concurrency level:** ~10 reports running simultaneously
- Based on the above information I would suggest the below tuning.
 - Set the number of Report Service processes to 19
 - Set the dispatcher jvm for report server to 4196mb
 - Set the dispatcher jvm for content manager to 8192mb
 - Tune the cgs process to 2gb, minimum (4gb as this is a large server.)
 - Set dynamic query to 8192mb once these types of reports start being used
 - Batch report service we need to determine how many batch reports are running and tune accordingly
- These are suggestions only. The system will need to be monitored to determine if these are the optimal settings