Multiple Garbage Collectors: Can Two Sanitation Engineers be Better Than One?

Posted At : November 29, 2007 1:48 PM | Posted By : Mark Kruger
Related Categories: Coldfusion Optimization, Coldfusion Troubleshooting

I ran across this JVM configuration on a server recently. I should note that the server in question was having some issues, so this is *not an endorsement* of this approach. I simply had not seen this sort of configuration on a Coldfusion server before. Here are the JVM arguments:

```
java.args=-server -Xms512m -Xmx768m -Dsun.io.useCanonCaches=false -XX:MaxPermSize=192m
-XX:PermSize=64m -XX:NewSize=48m -XX:+UseParNewGC -XX:+CMSParallelRemarkEnabled
-XX:+UseConcMarkSweepGC ...
```

What seemed unusual to me is that this particular set of arguments allows for 2 garbage collectors to be specified. Both the "UseParNewGC" switch and the "UseConcMarkSweepGC" switch are set. I did not know that multiple GCs could be specified. I set about finding out how this was possible.

If you have ever used the Coldfusion Administrator to change garbage collectors you may have run into the issue where Coldfusion adds the UseParallelGC garbage collector back into the arguments. For example, if you change the garbage collection to "UseConcMarkSweepGC" and then update (again - using the Coldfusion Administrator), you will end up with *both* the UseParallelGC *and* the UseConcMarkSweepGC as arguments to the JVM. When you restart Coldfusion it will fail - complaining that it cannot handle 2 separate GC arguments. See this **previous post** for a more thorough discussion of that issue.

However, this does not mean that *all* garbage collectors are mutually exclusive. In my reading I missed that tidbit. The arguments above show how they can be made to live together. After a bit of googling I found several warnings that explicitly tell me that "UseConcMarkSweepGC" and "UsParallelGC" are indeed mutually exclusive. Since that is the error I ran into I suppose I jumped to the conclusion that GC's are mutually exclusive *in general*. Now I know that this is not always the case. Note also the CMSParallelRemarkEnabled switch in the example above. This switch is designed to reduce the pauses that occur when the GC is marking objects for deletion.

In my searching on the topic I found **this document** in the help files for CAMS (a network security policy server from **Cafesoft** which runs on Java). While some of the advice is specific to CAMS, I found the overview to be excellent. In particular there is some good information about the differences between multiprocessors and single processor machines - and why various garbage collectors work or do not work well with them.

I know there are some Java/Coldfusion gurus reading my blog. Perhaps you could chime in with some additional information.