

General Performance Recommendations

Version 2025



Contents

SIMULATOR OPTIONS	1
Using more than 64 threads	1
BIOS/UEFI	2
Adjust for Best Performance	2
Node Interleaving: Disable (NUMA On)	2
Sub NUMA Settings	3
Logical Processors (Hyper-threading)	3
Turbo Boost: Enable	3
Snoop Mode	3
Operating System	4
Environment Variables	4
Hardware	6
CPU	6
Memory	6
Disk	6
Network Associated Storage (NAS)	6
Additional Notes on KMP_AFFINITY	7
Hyper-Threading Recommendation	9
Notes on Microsoft HPC Scheduling Software	9

SIMULATOR OPTIONS

Using more than 64 threads

On hardware which has more than 64 cores it may be useful to use the following simulator options when using more than 64 threads (i.e. **-parasol 192**):

-pdegab 0 on the command line or ***PDEGAB 0** in the numerical section of the input data; reduces the connections and communication overhead between parallel partitions in solver.

-combinative ilu_auto on the command line or ***COMBINATIVE ILU_AUTO** in the numerical section of the input data; disables the AMG solver which may not perform as well using many cores.

BIOS/UEFI

Adjust for Best Performance

Most manufacturers have a "Best Performance" default profile that can be set in the BIOS. Start with setting the bios to the "Best Performance" default before moving to any other settings. On Dell[†] Systems there is a setting for Memory Optimization called "MemOpMode" - set it to *OptimizerMode*.

If you require additional BIOS settings information, please contact the manufacturer directly.

Node Interleaving: Disable (NUMA On)

NUMA (Non-Uniform Memory Access) is a computer memory design used in multiprocessing, where the memory access time depends on the memory location relative to the processor (*Figure 1*). Under NUMA, a processor can access its own local memory faster than non-local memory

(memory local to another processor or memory shared between processors). The benefits of NUMA are limited to particular workloads, notably on servers where the data are often associated strongly with certain tasks or users*.

Excerpt from: https://en.wikipedia.org/wiki/Non-uniform_memory_access

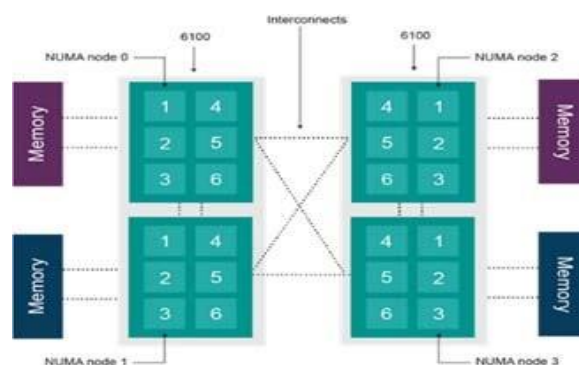


Figure 1: Non-Uniform Memory Access (NUMA)

For older (Intel) processors, depending on the system used, this setting may have multiple names; however, NUMA should be on. On most Intel[†] systems, this setting will include the word "interleaving", and will need to be disabled. Confirm the name and setting details in the hardware manufacturer's documentation.

When NUMA is enabled, the system tracks which memory is "local" to a CPU and will take advantage of faster access times inherent with local memory (hence "non-uniform access"). Otherwise, all memory is treated as non-local and no performance gains are realized.

Sub NUMA Settings

With modern Intel and AMD processors such as Ice Lake and Milan respectively, Sub NUMA settings control how groups of cores within an individual processor are handled. For Ice Lake Xeon Gold 6338, for instance, the processor can be logically split into two NUMA nodes with 16 cores each in the BIOS on some systems by setting "Sub NUMA Cluster" to "2-Way Clustering" which will treat each processor as 2 NUMA nodes, or to "Disable" to disable it. For Milan EPYC 7513, which consists of four 8-core chiplets, systems may have "L3 cache as NUMA Domain" "Enable" which cause the operating system to see the processor as 4 NUMA nodes or set to "Disable" for the operating system to see each processor as a single NUMA node.

For both of the above, it is recommended to disable the above settings so that each processor is seen as a single NUMA node for a total of two NUMA nodes on a two socket system.

Logical Processors (Hyper-threading)

Previous releases of CMG simulators recommended that hyper-threading be disabled. This is no longer required except on Windows machines.

Turbo Boost: Enable

To maximize performance, turbo boost allows the processor to run at a faster clock rate than its base frequency, if requested by the Operating System. The increased clock rate is limited by the number of active cores, the estimated current and power consumption, and the processor temperature.

Snoop Mode

On multi-socket machines, the Snoop Mode setting determines how processors monitor changes made to memory within other processors cache lines. This can be important for CMG software performance. CMG recommends using "Opportunistic Snoop" mode where available.

Note: There are several other options that can potentially impact performance, such as C-State and Monitor/MWait. Implementations can vary, so please review the vendor's recommendations for best practices and other related settings to optimize CPU performance.

Operating System

In general, anything that improves the overall performance of the Operating System (e.g. disabling unnecessary processes) will also improve CMG software performance. Performance tuning, which increases the performance of one aspect of the OS or hardware at the expense of another, can help but should be carefully tracked and tested. Please note, CMG cannot make specific recommendations, as there are too many variables (different hardware, OS versions, network environments, etc...). There are many available resources, such as the white paper "Performance Tuning Guidelines for Windows[†] Server" or the "RHEL[†] Performance Tuning Guide" for those particular Operating Systems.

Environment Variables

Beyond changes to your BIOS, there are environment variables that affect how the CMG simulators interact with your hardware.

OMP_PLACES tells the Operating System where to place execution threads from the simulator. Set **OMP_PLACES=cores** to bind threads to cores to take advantage of cached information in "local" memory for that core.

KMP_AFFINITY and OMP_PROC_BIND tell the Operating System which specific places to bind threads to. As a general guideline we recommend letting the OS choose. Previously the recommendation to set KMP_AFFINITY as follows may work well for machines with 64 or fewer cores:

3 rd Party Scheduler? (E.g. LSF, HPC)	Number of Jobs Run Simultaneously	All Cores Used?	Setting
Yes	1	n/a	KMP_AFFINITY=compact,1 [*]
Yes	2+	n/a	KMP_AFFINITY=compact,0
No	1	yes	KMP_AFFINITY=compact,0
No	1	no	KMP_AFFINITY=compact,1 [*]
No	2+	n/a	Do not set

KMP_AFFINITY has to be manually adjusted per job if multiple jobs are running on the same machine, outside of a scheduling software such as HPC or LSF. CMG does not recommend this scenario.

The KMP_AFFINITY recommendations outlined above hold regardless of whether hyper-threading is enabled or disabled.

OMP_SCHEDULE affects the execution of loops within the program based on number of threads available. For IMEX™ and STARS™ set OMP_SCHEDULE=static,1. This setting is not necessary for GEM™, as it is managed internally.

When experimenting with these variables it is a good idea to set **OMP_DISPLAY_ENV=true** so that OpenMP will print out at runtime how it has interpreted the specification. Also set **OMP_DISPLAY_AFFINITY=true** to display formatted affinity information for all OpenMP threads.

*For some cases on Ice Lake and on Milan on Linux, running a single job using less than 64 cores and reserving the whole server for the job, and not specifying KMP_AFFINITY nor OMP_SCHEDULE can improve run time. Using the default for KMP_AFFINITY and OMP_SCHEDULE setting for Ice Lake tended to be better for most cases when running a single job using less than 64 cores on a variety of IMEX problems, with an elapsed time of 200 seconds or more, on a dual socket Ice Lake Xeon Gold 6338 server with RHEL 8.4. On dual socket Milan EPYC 7513 for problems with less than 1 million active cells, running compact,0 tended to be better while for larger cases, using the default settings for KMP_AFFINITY and OMP_SCHEDULE tended to be better.

The functionality of these settings can be hardware dependent; consequently, the optimal setting for a user's specific setup may be different than above. These are only general guidelines and should be tested in any new environment.

Hardware

Note: these are only general guidelines only. Please contact CMG Sales for a detailed description of the hardware recommended/used internally at CMG.

CPU

CMG software is very CPU-intensive and we always recommend CPUs with the best available combination of clock speed and cache. The optimal number of cores is dependent on the types of jobs, number of users, licensing, and other factors; talk to CMG Sales to determine the optimal number for the specific situation.

On dual socket node with Intel Ice Lake Xeon Gold 6338 vs. AMD Milan EPYC 7513 processors, both of these perform well, and both have 32 cores per processor for a total of 64 cores per server. Using the above settings for KMP_AFFINITY and OMP_SCHEDULE for a single job and reserving the whole machine for the job, Ice Lake tended to run faster. Running simultaneous jobs, using KMP_AFFINITY=compact,0; OMP_SCHEDULE=static,1 tended to be faster and Ice Lake tended to be faster than Milan for IMEX jobs with 200 elapsed time or more.

Memory

Memory speed is an important factor in CMG performance and should be the fastest speed available from the hardware vendor for a specific machine. During internal testing, CMG has seen significant slowdowns with "low-voltage" RAM, which is occasionally found in lightweight laptops. In addition, it is important that memory is properly balanced to take advantage of this speed. For example, Ice Lake and Milan processors, 8 DIMMs/CPU is the recommended configuration that can be used to achieve the best performance. If decreased to lower than 6 DIMMs/channel, the speed drops. There are several downloadable tools, such as CPU-Z, to confirm the most effective memory speed.

Page file should be set to a fixed size rather than automatically managed by the Operating System; this eliminates the overhead of managing the page file and prevents fragmentation.

Disk

Disk I/O is generally not a bottleneck for CMG software. However, if disk I/O is possibly affecting performance, this can be easily tested by monitoring performance on the server using a number of tools. Perfmon (Performance Monitor) in Windows[†] or iotop in Linux[†] are two examples, but any standard monitoring tool should work.

Network Associated Storage (NAS)

For improving NAS performance, please consider the following options:

- Make sure the device is dedicated for high performance file accessing and disable unnecessary features or protocols, especially compression, deduplication, logging, snapshots, or other scheduled tasks.

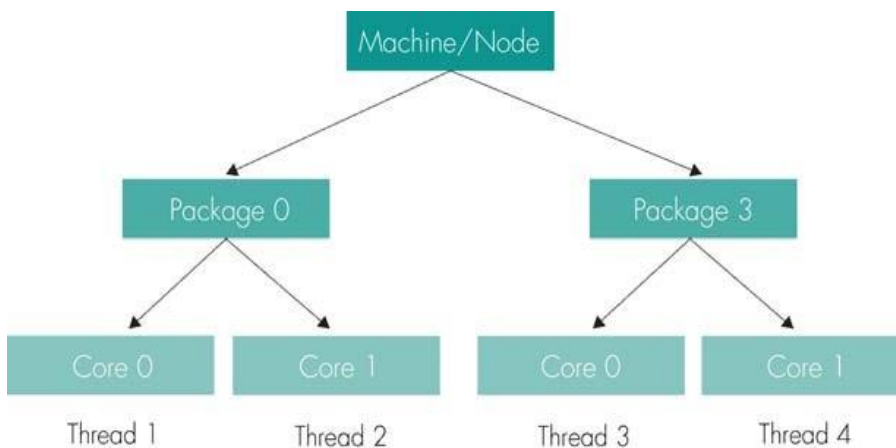
- Use a high-speed link dedicated only for HPC nodes (no other traffic); CMG prefers a link speed of 10GB or higher.
- Consider adding more memory and using high end processors. Some file systems require at least 16GB Memory. Adding memory or SSD cache can increase NAS performance.

Test job performance on compute node using the local disk and compare the performance with the same job run on the NAS to see if there is any degradation in performance.

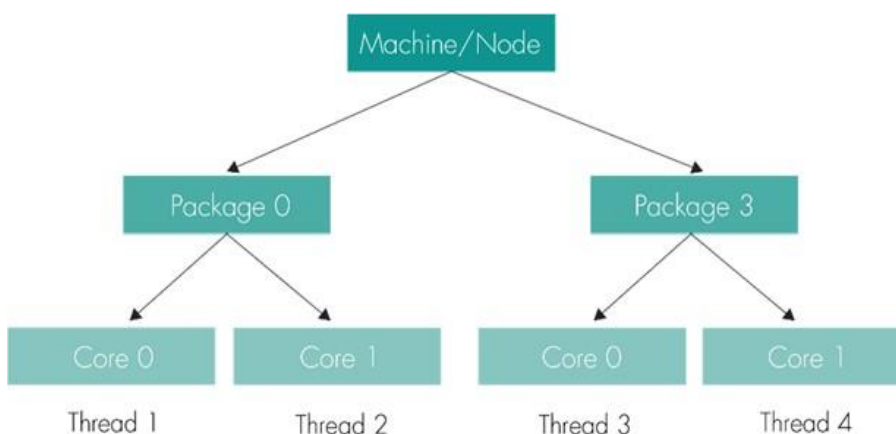
Additional Notes on KMP_AFFINITY

When CMG simulator jobs are run outside of a job scheduler and the KMP_AFFINITY environment variable is set, it is possible that concurrent jobs may compete for the same cores. For example: If KMP_AFFINITY is set to compact,0 without a job scheduler and run two concurrent jobs, both jobs will start on core 0 on the first socket. Job schedulers usually know which cores are available and will start the first job on the first core and the second job on the first core that is not assigned to the first job to avoid over-subscription of resources.

For example: using a Nehalem-EX 8 Core, 4 Socket Server (32Cores) and specify "KMP_AFFINITY=compact,0", use a job scheduler and run four 8-thread jobs. The first job will use all the cores on the first socket (8), the second job will use all the cores on the second socket (8), the third will use all the cores on the third socket (8) and the fourth will use all the cores on the fourth socket (8), as shown in the figures below.



For the same case, using `KMP_AFFINITY=compact,1`, the first thread of the first job will run on core0 of the first socket, the second thread of the first job will run on core0 of the second socket and so on. This causes memory traffic between the sockets and is not recommended. However, this may be the best setting for single jobs, on a system that allows overclocking (turbo mode) to optimize memory usage.



The recommended configuration is to set `KMP_AFFINITY=compact,0` on all nodes of the clusters and ALWAYS use HPC job scheduler for CMG.

This variable setting is becoming more important due to the newer multi-core systems now available and needs further investigation. The proper setting for this variable depends upon the type of simulator, number of cores required for job and number of jobs per system being run simultaneously.

Hyper-Threading Recommendation

For optimal performance, releases prior to 2018.10 recommended that hyper-threading be disabled. This is no longer required, with the following considerations:

1. Setting KMP_AFFINITY to non-default values is no longer required. If running a single job while reserving the whole server for a dual socket Ice Lake Xeon Gold 6338 or Milan EPYC 7513 with RHEL 8.4, then using default settings may provide a benefit. Please set "KMP_AFFINITY=" to the default setting and use "OMP_PROC_BIND=CLOSE" or "OMP_PROC_BIND=SPREAD" instead for older AMD processors.
2. Extensive testing, using Intel processors, has shown that better performance is achieved when hyper-threads are not used. This is now the default simulator behavior for Linux which means that the number of threads (requested for the simulation) cannot exceed the number of physical cores, unless the command line option '-htuse' is used. GEM also allows use of keyword *HTUSE *ON in the data file.
3. Furthermore, when the number of physical + hyper-thread cores in a machine is greater than 64, the use of Linux_x64 executable is strongly recommended; the affinity setting is not effective (at more than 64 cores) on numerous Windows OS variants.
4. The keyword has no effect on machines where hyper-threading is off. It also has no effect on machines with operating systems that do not respond to thread-binding by setting KMP_AFFINITY.

Hyper-threading effects could depend on processor type, hardware configuration and number of jobs scheduled per node. Please consult your IT department and/or CMG support for further guidance.

Notes on Microsoft HPC Scheduling Software

In our limited testing, we had a performance issue with the Cascade Lake class of Xeon dual processor nodes and Microsoft HPC 2016 Scheduling Software. Microsoft HPC 2019 Scheduling Software along with RHEL 8.4 and a BIOS update appears to resolve the issue. In some cases, using HPC 2016, performance may improve by reserving a core per socket for overhead related issues or by rebooting.

CMG General Performance Recommendations
2025

[™] Trademark of Computer Modelling Group Ltd. Copyright © 2025 Computer Modelling Group Ltd.
* Other company, product, and service marks are the properties of their respective owners.