CMAQ RUNTIME PERFORMANCE AS AFFECTED BY NUMBER OF PROCESSORS AND NFS WRITES

Patricia A. Bresnahan,[a]* Ahmed Ibrahim [b], Jesse Bash [a] and David Miller [a]
[a] University of Connecticut, Dept. Natural Resource Management and Engineering, Storrs, CT
[b] University of Connecticut, Dept. Computer Science and Engineering, Storrs, CT
Email: pbresnah@canr.cag.uconn.edu
Voice: (860)486-2840 FAX (860)486-5408

1. INTRODUCTION*

The recent release of a parallel version of the Models3 CMAQ code (Byun and Ching, 1999) has prompted many research labs to migrate to multi-processor environments such as Linux clusters.

One question that needs to be addressed during the requirements analysis phase of migration concerns the number of processors that should be acquired. While it is generally agreed that better processing times can be achieved in multi-processor environments, the optimal number of processors for a given application depends on a number of factors such as the parallelization method, the size of the datasets and the data transfer limitations between processors.

The purpose of this paper is to describe how processing time for the parallel version of CMAQ was affected by the number of processors used, local vs. remote data access and the size of the datasets involved.

2. HARDWARE CONFIGURATION

The Atmospheric Resources Lab (ARL) at UCONN (University of Connecticut) recently purchased a Linux cluster. This consists of one head node, four slave nodes and a file server as shown in Figure 1. The details of the components are shown in Table 1 below.

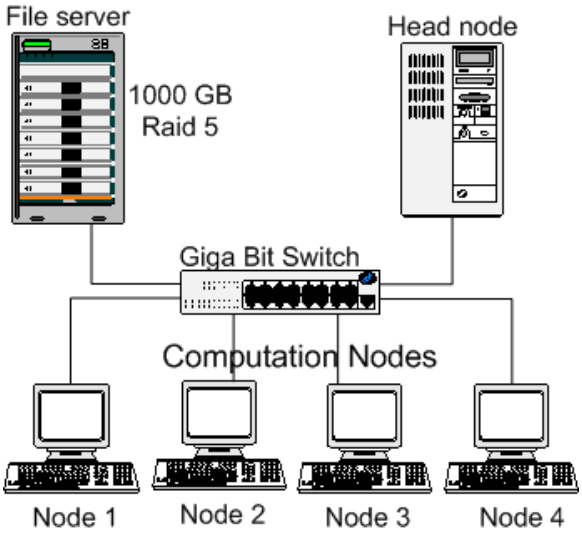


Figure 1. Configuration of the UCONN ARL Linux Cluster.

3. SOFTWARE

The cluster operating system is RedHat Linux 7.3, with the parallel libraries provided by MPICH version 1.2.5.

The May 2003 release of CMAQ (version 4.2.2) and other Models3 tools (netCDF, IOAPI, MCIP) were compiled and the resulting executables used for this test.

*Corresponding author address: Patricia A. Bresnahan, Dept. Natural Resource Management and Engineering, University of Connecticut, Storrs, CT, 06269-4087

Table 1. UCONN ARL Linux Cluster Description
1) Head Node: • Dual CPU's AMD MP 2000, 1.667GHz CPU's, 266Mhz Bus, 2.048GB PC2100 ECC RAM • (2) 120GB EIDE Hard Drives, 7200 RPM, Mirrored RAID level 1
2) 4 Slave Nodes: • Dual CPU's AMD MP 2000, 1.667GHz CPU's, 266Mhz Bus, 2.048GB PC2100 ECC REG DDR RAM • 120GB EIDE Hard Drive
3) EIDE RAID 5 File Server: • AMD XP 2000, 512 MB DDR PC2100 RAM • (1) 20GB Hard Drive • (8) 120GB Hard Drive
4) 8 Port Gigabit networking Switch
5) Operating system: Linux

4. DATASETS

Two datasets were used in this study. A "small" spatial domain, consisting of the tutorial files provided with the Models3 tools (38x38x6, 24Hrs) and a "large" spatial domain created in our lab based on an MM5 run provided to us through NYDEC, that was generated at the University of Maryland. This dataset is based on the 36km Unified Grid and covers the eastern portion of the United States (67x78x21, 24Hrs). Table 2 shows a summary of the two domains, including the sizes of some of the largest input and output files.

Table 2. Description of Datasets		
Feature	**"Small" Tutorial**	**"Large" CT ARL**
Rows	38	67
Columns	38	78
Layers	6	21
Timesteps	24	24
In: Emissions (bytes)	37,383,992	37,543,856 (area only)
In: MetCRO3D (bytes)	34,322,976	634,358,520
Out: CCTM_*CONC (bytes)	67,624,592	821,822,876

It should be noted that the emissions file generated for the larger domain only contained area source emissions.

5. METHODS

The experimental design looked at runtime as affected by three factors: number of processors, dataset size and data IO location. A total of 48 runs were made (see Table 3).

Table 3. Experimental Design: Runtime Configurations				
Number of Processors	**Local Read / Local Write**	**Local Read / Remote Write**	**Remote Read / Local Write**	**Remote Read / Remote Write**
1	• Small • Large	• Small • Large	• Small • Large	• Small • Large
2	• Small • Large	• Small • Large	• Small • Large	• Small • Large
4	• Small • Large	• Small • Large	• Small • Large	• Small • Large
6	• Small • Large	• Small • Large	• Small • Large	• Small • Large
8	• Small • Large	• Small • Large	• Small • Large	• Small • Large
10	• Small • Large	• Small • Large	• Small • Large	• Small • Large

Number of Processors: For the single processor runs, the parallel version of the CMAQ code was not used. For the multi-processor runs, the parallel version was used and the run script modified as needed for the correct number of processors.

Data IO: For "Local Read" runs, all input files were stored on and read from the head node, while for "Remote Read" runs, these same files were stored and read from the file server. For "Local Write" runs, all output files were written out to the head node, while for "Remote Write" runs, these same files were written out to the file server.

Runs were made sequentially, in that a run was not started until the previous run had finished. No other work was being done on the cluster at the time of the runs.

Timing statistics were collected from the log files, which store the output of the UNIX "time"

command. The total amount of time spent in by the computer in "user" mode, "system" mode, and the total elapsed time was recorded for each run.

6. RESULTS

Results are shown in Figure 2 below. Figures 2a and 2d show that for both the large and small datasets, and for all data IO locations, the amount of time spent in user mode decreased as the number of processors increased. User time is spent on computational tasks that are not related to data IO. It is the number of seconds the process got CPU cycles to be processed. The greatest decrease in processing time occurred as the number of processors increased from one to four. Beyond four processors, there were still decreases in time, but not as great.

System time is spent on tasks related to data IO and other "housekeeping." It is the number of seconds the system (kernel) spends on behalf of the process. Figures 2b and 2e show that when the data IO involved Remote Writes, increasing the number of processors actually resulted in an increase in the amount of time spent in "System" mode. Reading remotely did not seem to add to the amount of system time used, however, regardless of the number of processors.

Elapsed time is the real 'wall clock' time, or the finish-time minus the start-time. Figures 2c and 2f show the total elapsed time of each run. When data was written locally, run time decreased as processors were added, especially as the number of processors went from one to four. When data was written remotely, however, processing time actually increased as the number of processors exceeded two, for the small dataset, and four for the large dataset.

7. DISCUSSION AND CONCLUSIONS

In general, CMAQ ran faster as more processors were used, as long as the data was written locally. When data was written remotely to the file server, adding processors actually increased processing time due to the greater overhead involved in the remote writes.

In the LINUX cluster environment, these remote writes are accomplished through calls to NFS (Network File System) write. This call involves a fair amount of system overhead to ensure data integrity. As Juszczak (1994) states: "…the protocol requires that data modification operations such as write be fully committed to stable storage before replying to the client. The cost of this is significant in terms of response latency and server CPU and I/O loading." NFS write performance can be slightly tuned-up through adjusting some performance parameters such as block size and transfer protocol. However, read performance of NFS is good and comparable to local disk read.

As of this writing we have not tested version 4.3 of CMAQ (released in September 2003), but noticed that NFS latency problems in an MPICH cluster appear to have been addressed. We plan to repeat our test on this latest release sometime during the next month.

8. REFERENCES

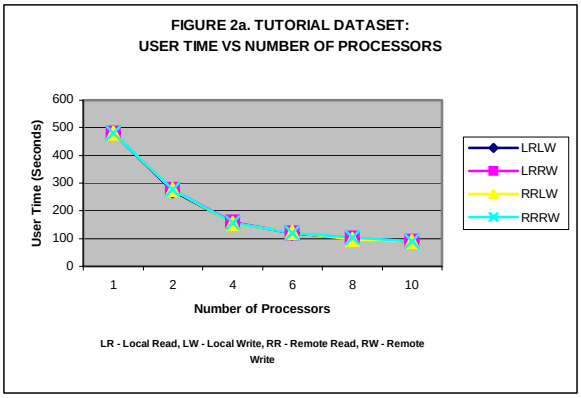
Byun, D.W. and J.K.S. Ching. 1999. Science Algorithms of the EPA Models-3 Community Multiscale Air Quality Modeling System. EPA/600/R-99/030. Washington, D.C. United States Department of Environmental Protection, Office of Research and Development.

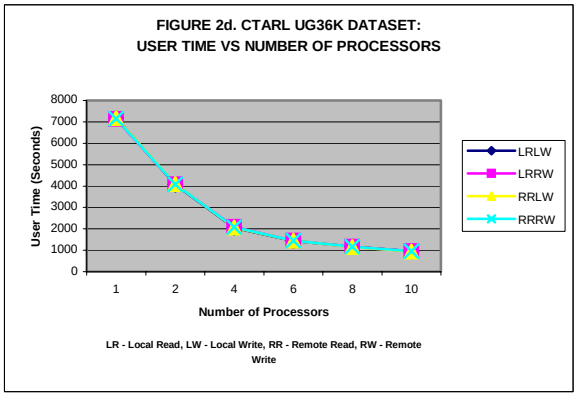
Juszczak, C. 1994: Improving the Write Performance of an NFS Server. *Proceedings of the USENIZ Winter 1994 Technical Conference*, San Francisco, 247-259.

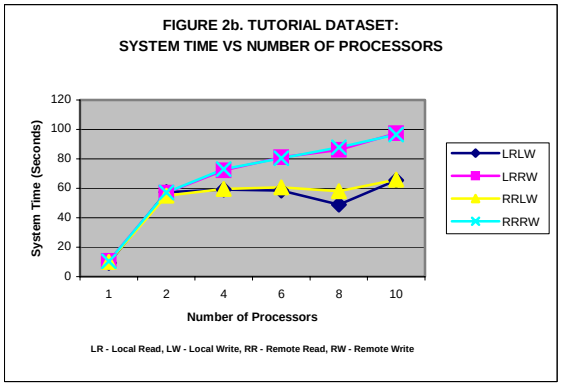
9. ACKNOWLEDGEMENTS

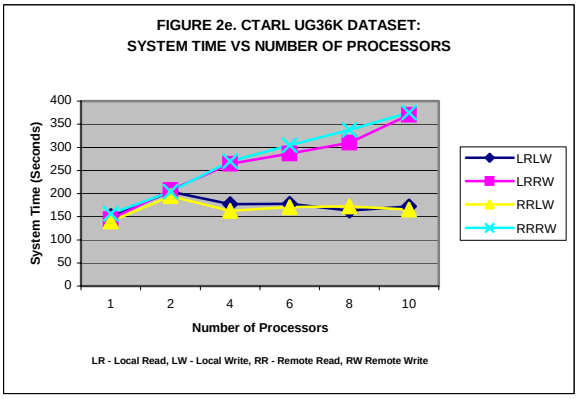
We would like to thank the NYDEC and the University of Maryland for the use of their MM5 1997/2002 dataset.

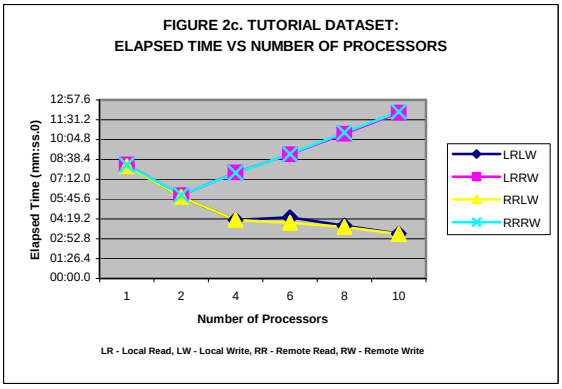
This research was supported by the Connecticut River Airshed Watershed Consortium, USEPA Cooperative Agreement R-83058601-0.











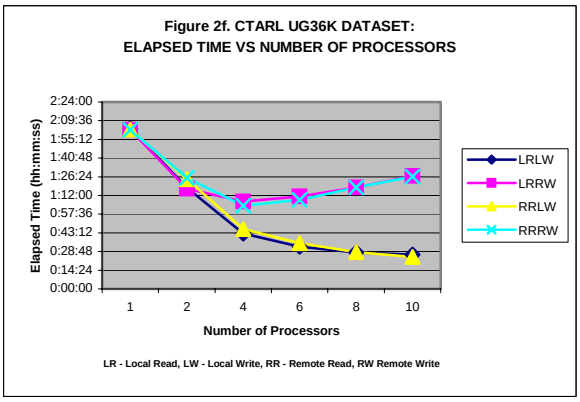


Figure 2. CMAQ runtime performance as affected by number of processors, data IO location and dataset size.