GAMESS Patch for SGI Origin (GDDI-O3K)

Package ID: GDDI-O3K-1a-22Feb2006R2
Release date: Aug. 7, 2006

Kiyotaka Suzuki, Ph.D.
<ksuzuki@bri.niigata-u.ac.jp>

Department of Biological Magnetic Resonance
Center for Integrated Human Brain Science
University of Niigata
1-757 Asahimachi, Niigata, 951-8585 Japan
http://coe.bri.niigata-u.ac.jp/

1. Brief description of the software patch

GDDI-O3K is a patch suite against GAMESS[1-3], a general *ab initio* quantum chemistry package maintained by the members of the Gordon research group at Iowa State University, that provides users of SGI Origin with access to the functionality of "processor groups". This functionality is essential for performing analyses based on the fragment molecular orbital method (FMO)[4, 5] and is therefore particularly important for analyzing systems that contain large molecules such as polymers and proteins and/or a large number of small molecules such as the water cluster. Unfortunately, however, it has so far been unavailable on parallel computers that utilize Shared Memory Access (SHMEM) as its low level inter-node communication protocol.

Most of our modifications are made to the distributed data interface (DDI) portion of the code. DDI is the message passing layer on which the application part of GAMESS is constructed,[6-8] and is designed to provide a universally applicable model of processing very large arrays whose patches are distributed among CPU nodes. GAMESS developers set one of their goals as to make it run on any type of Unix computer, from a cluster built from symmetric multi-processor (SMP) enclosures to high-end massively parallel computers. Actually almost all types of UNIX machines have complete access over every functionality of the previous (second) generation of DDI[7] regardless of the type of the communication model: TCP/IP sockets, MPI, LAPI (of IBM), or SHMEM.

At the release of the latest (third) generation of DDI, a concept of dynamic grouping of processors was newly incorporated.[8] Under the scheme, processors can be grouped into subsets

working on completely independent quantum mechanical tasks. This extended DDI model embodies hierarchical parallelization and is termed "generalized DDI (GDDI)". Its implementation has, however, been excluded in isolated codes for SHMEM machines, namely, `ddio3k.src` and `ddishmem.src`, and consequently users of Cray X1/T3E, Compaq SC, and SGI Origin/Altix have never benefited from the significant improvement in DDI. We guess that this situation might not be for technical reasons, but being due to a limited number of contributors whose recent efforts have naturally focused on tuning to symmetric multi-processor (SMP) clusters, which are the most commonplace parallel computer system today.

The following reasons made us decide to modify the DDI code by ourselves: (1) We found that our analyses would need to use FMO, and SGI Origin is our main machine. (2) It seemed that one of the future improvements in DDI would be increased scalability based on use of one-sided communication routines such as those of SHMEM and MPI-2. In our modifications, all SHMEM operations in the dedicated FORTRAN code (`ddio3k.src`) have been successfully merged into the standard source files written in C.

2. Disclaimer

Our software patch GDDI-O3K is provided on an "as is" basis, and the author and Center for Integrated Human Brain Science (CIHBS) disclaim any and all warranties, either expressed or implied, with regard to this patch. Download and/or use of the patch and accompanied files are expressly at users own risk. Please do not ask the GAMESS developers about problems you have with our patch.

3. Update history

Aug. 7, 2006 Released the first version (GDDI-O3K-1a-22Feb2006R2)

4. Who will be benefited from our library?

Our patch is particularly useful for Origin users who feel unsatisfactory about the lack of capability to run FMO jobs on the system. We think that the modified DDI library will also be able to run on SGI Altix with a small change in a few files. Martin Hilgeman, an application engineer at SGI Netherland, kindly told me that System V (SysV) shared memory + TCP/IP sockets DDI

implementation, which is now the default in GAMESS, is not the most efficient for systems with lots of processors, and that this is not due to the way it is implemented in GAMESS, but rather a drawback of SysV shared memory. SGI SHMEM provides much lower latency, higher bandwidth, and higher scalability compared to SysV + sockets by making the maximum use of NUMAlink, a sophisticated system memory interconnect of Origin and Altix. Therefore, for either of these machines, we have no reason to choose something other than SHMEM as the communication layer of DDI. We hope that our patch will help GAMESS developers to take the SHMEM code back to the main track of development.

5. Installation

The current patch should only be applied to version 22Feb2006(R2) of GAMESS.

(1) Preparation

If you have GAMESS already installed in your directory, save the directory tree as a different name in case of an accidental loss of the contents (e.g. by "cp –r gamess gamess.org"). If you have never installed GAMESS version 22Feb2006(R2), or if you have modified the original code, you should first put the original code somewhere in your directory. The GAMESS package is freely available at http://www.msg.ameslab.gov/GAMESS/GAMESS.html.

(2) Downloading the patch

Click on the file names below to download our patch and a shell script for installing the patch:

http://coe.bri.niigata-u.ac.jp/coedoc/ScriptDownload/contents/gddi-o3k-1a-22Feb2006R2.patch

http://coe.bri.niigata-u.ac.jp/coedoc/ScriptDownload/contents/applypatch-1a.sh

These files should be put in the directory where the top of your GAMESS directory resides (e.g. ~/gamess). The patch includes, in addition to extensive changes for implementing the GDDI functionality, small modifications to several files for successful compilation and execution of GAMESS on IRIX, partly according to a patch for an older version of GAMESS, which was created by Martin Hilgeman at SGI. (His patch was formerly available at the web site of SGI, but it seems that the web page is currently closed.)

(3) Applying the patch

Once you make sure that your own GAMESS directory has been saved, you may apply our patch by executing `applypatch-1a.sh`. If you get an error of "permission denied", just do "`chmod 755 applypatch-1a.sh`" and try again. A list of files that will be changed by the patch is given in Chapter 7.

(4) Making the DDI library

First look into gamess/ddi/`compddi` and modify it for your system. Running compddi will then generate the modified library with the same name as the original (libddi.a) unless you have errors. Note that the script has been changed to invoke C compiler rather than FORTRAN, which is quite unlike the original one, and that DDI_SHMEM is newly added to the command-line options to activate SGI SHMEM codes in the DDI source files.

(5) Building GAMESS

Modify `comp`, `compall` and `lked` placed in the top of the GAMESS directory tree so as to meet your system environment. Refer to `PROG.DOC` for general instructions on how to build a GAMESS executable file.

6. Running FMO jobs

The script to run a GAMESS job, `rungms`, has also been a bit changed. Take a look and modify it for your system environment. Our modification affects in particular the behavior of displaying the progress of calculation when the `$gddi` option is activated in an input card and the number of groups (`ngroup`) is set larget than 1. In such a case, outputs of every process will be written to respective files, $SCR/`JOB.F06`, $SCR/`JOB.F06.001`, $SCR/`JOB.F06.002`, etc., where $SCR represents the scratch directory (e.g. /work/scr/$USER), and the progress of calculation will not be displayed in the shell window from which a GAMESS job is invoked.

For convenience, we provide a utility script to monitor the contents of the log files on X-window environment. The script file is named "`gddimon`" and located at gamess/ddi/tools/sgi. Before using it, change the definition of the scratch directory so as to meet your system and put the script file somewhere in your search path. Once a GDDI job is launched with an input card named `JOB.inp`, you can start monitoring the progress whenever you like by executing a command "gddimon JOB" in an appropriate shell window. This will constantly show you outputs of the global master process in a new window. If you want to monitor some other processes besides the global master, specify the ranks of target processes after the job name, e.g. "gddimon JOB 1 2 3". This will open multiple windows as much as the number of the target processes. It is usually better

to select only group masters, which are exclusively chatty.

Another useful script added by our patch is `rstprep`, which can be utilized to duplicate an F40 (or F30) restart file. The modified `rungms` will save restart files in ~/scr. In a case where `~/scr/JOB1.F40` is required by a restart job JOB2 that will invoke *N* processes, for example, `JOB1.F40` needs to be copied in the scratch directory $SCR as a series of files `JOB2.F40.000`, `JOB2.F40.001`, ···, `JOB2.F40.`*N*-1 in advance of running JOB2. You can do this easily by "rstprep ~/scr/JOB1.F40 $SCR/JOB2.F40 *N*". Note that, although FMO in GAMESS allows selecting "broadcast of F40" (with 64 added to `MODPAR`), it requires the F40 file NOT to exist in the scratch directory of slave nodes, and is obviously not applicable to Origin/Altix, on which the directory is visible to all computational nodes. For the same reason, `NODEXT` should be set to zero for any FMO job.

We found that the script `grabcube` in gamess/tools/fmo/fmocube, which is a utility to extract Gaussian-like cube from GAMESS punch files, does not work on systems that do not have GNUish '`grep`' and '`tail`' installed. When invoked as standard UNIX commands, `grep` does not accept the "m -1" option, while `tail` saves the last part of a file in an internal buffer. The output of tail is thus limited in length, which seems not enough for generating complete cube data even with a common grid size. Accordingly, we included a file `grabcube2.c` in our patch as a substitute for the script. This is a simple C file and you can make the executable by "`cc –o grabcube2 grabcube2.c`". Its usage is exactly the same as that of `grabcube`.

7. Overview of our modifications

SGI Origin/Altix is a cache-coherent distributed shared memory (DSM) machine and therefore intrinsically fits for the fundamental concept of DDI. The shared address space can be explicitly accessed via SHMEM operations. A framework of SHMEM-based DDI has been established in the dedicated code `ddio3k.src`, and is now entirely merged into the standard source codes located in gamess/ddi/src. As a result of converting it to C, the problem of ending up with INTEGER*8 variables defined in the system header files `mpif.h` and `shmem.fh` is avoided. Increased flexibility in managing address pointers is another merit of using C.

We owe the implementation of the GDDI functionality to a complementary use of SHMEM and MPI. While SHMEM routines minimize the overhead associated with requests for data transfer, maximize bandwidth, and minimize latency, its way of splitting up the communication world is a bit inflexible. Utilization of MPI routines makes it easier to realize dynamic grouping of processes. In exchange for the GDDI capability, however, we decided to abandon the "truly" dynamic memory allocation for distributed arrays, which has been specifically employed in the SHMEM

version of DDI. On IRIX, a pool of memory that is accessible from other processor nodes must be allocated from a symmetric heap. In other words, allocating such a shared region is a global task on Origin and is inconsistent with intra-group allocation in nature. Nevertheless, our decision to give up "allocation on demand" is nothing less than to follow the original way of GAMESS in which an aggregate amount of the shared memory pool required throughout the execution of a job is allocated at a very early phase of run.

Our GAMESS patch was intensively tested on an Origin 3800 system running IRIX 6.5.27m, MPT 1.9, and MIPSpro 7.4 compilers. All of the following test runs completed in a successful manner: sample FMO jobs located in gamess/tools/fmo/samples, a larger FMO job (gamess/tools/fmo/fmoutil/1crn1res.inp.sample), and jobs quinone.inp and si9h12.inp that Dr. Mike Schmidt of the Gordon Research Group kindly sent to us for testing everything about distributed memory. GAMESS source files that are modified by our patch are as follows:

Directory	Files that will be modified			
gamess	comp	compall	lked	rungms
gamess/source	prppop.src	unport.src		
gamess/tools/fmo/fmocube	grabcube2.c(*)			
gamess/tools/sgi(*)	gddimon(*)	rstprep(*)		
gamess/ddi	compddi			
gamess/ddi/include	ddi_base.h	mysystem.h		
gamess/ddi/src	ddi.c	ddi_acc.c	ddi_create.c	ddi_destroy.c
	ddi_dlb.c	ddi_finalize.c	ddi_gdlb.c	ddi_get.c
	ddi_getacc.c	ddi_group.c	ddi_index.c	ddi_init.c
	ddi_memory.c	ddi_put.c	ddi_recv.c	ddi_send.c
	ddi_shmem.c(*)	ddi_sync.c		

* indicates that the file will be newly added.

8. Future development

We are planning to replace SHMEM calls with MPI-2 functions. This is just for portability issues; it is believed that the performance of parallel jobs running on an SMP cluster will be also improved by use of one-sided remote memory access, as long as MPI implementation is optimized for the system.

9. Aknowledgements

We are grateful to Drs. Mike Schmidt, F. L. Gu, Dmitri G. Fedorov, and Martin Hilgeman for their assistance and expertise.

10. References

[1] M.W. Schmidt, K.K. Baldridge, J.A. Boatz, S.T. Elbert, M.S. Gordon, J.H. Jensen, S. Koseki, N. Matsunaga, K.A. Nguyen, S.J. Su, T.L. Windus, M. Dupuis, J.A. Montgomery, General atomic and molecular electronic structure system, J. Comput. Chem. 14 (1993) 1347-1363.

[2] M.S. Gordon, M.W. Schmidt, Advances in electronic structure theory: GAMESS a decade later, in: Theory and Applications of Computational Chemistry, the first forty years, Elsevier, Amsterdam, 2005.

[3] http://www.msg.ameslab.gov/GAMESS/GAMESS.html

[4] K. Kitaura, E. Ikeo, T. Asada, T. Nakano, M. Uebayasi, Fragment molecular orbital method: an approximate computational method for large molecules, Chem. Phys. Lett. 313 (1999) 701-706.

[5] D.G. Fedorov, K. Kitaura, The importance of three-body terms in the fragment molecular orbital method, J. Chem. Phys. 120 (2004) 6832-6840.

[6] G.D. Fletcher, M.W. Schmidt, B.M. Bode, M.S. Gordon, The distributed data interface in GAMESS, Comput. Phys. Commun. 128 (2000) 190-200.

[7] R.M. Olson, M.W. Schmidt, M.S. Gordon, A.P. Rendell, Enabling the efficient use of SMP clusters: the GAMESS/DDI model, in: Proc. of Supercomputing 2003, IEEE Computer Society.

[8] D.G. Fedorov, R.M. Olson, K. Kitaura, M.S. Gordon, S. Koseki, A new hierarchical parallelization scheme: generalized distributed data interface (GDDI), and an application to the fragment molecular orbital method (FMO), J. Comput. Chem. 25 (2004) 872-880.