IOPanda

Jonghyun Lee

Date: 2002/06/04 12:51:27

Abstract

Thorn **IOPanda** provides parallel I/O methods for 3D output of grid functions and multidimensional arrays in using the Panda parallel I/O library.

1 What is Panda?

Panda is a parallel I/O library developed at Professor Marianne Winslett's group in the University of Illinois, Urbana-Champaign. It is a easy-to-use, portable, high-performance I/O library which targets distributed and distributed-shared memory multiprocessors such as supercomputers and clusters of workstations. Panda supports parallel I/O for large-scale multidimensional arrays, which are the most frequently used data structure in scientific and engineering applications. It provides high-level array I/O interface for single-program multiple-data (SPMD) style programs. I/O operations supported by Panda are collective, where all the compute processors cooperatively perform I/O. Array snapshot and checkpoint/restart operations are supported in Panda. More information on Panda can be found at `http://drl.cs.uiuc.edu/panda`.

2 Why might you want to use Panda?

First, Panda provides high-performance collective array I/O. High-performance I/O can be achieved in Panda by utilizing I/O parallelism and Panda's unique I/O architecture called server-directed I/O. In server-directed I/O, I/O processors actively direct the entire I/O process rather than just passively respond to file read and write requests from the compute processors.

Second, Panda supports flexible in-memory and on-disk layouts. In Panda, applications can use different in-memory and on-disk layouts. If the user specifies different layouts in memory and on disk, Panda automatically and efficiently reorganize the data at I/O time. It supports both regular and irregular distribution of arrays. For regular distribution, Panda follows High-Performance Fortran (HPF) style distribution. The **IOPanda** thorn use (BLOCK, *, *) distribution as a default when writing out grid functions in Cactus, which makes the process to create a single array file much easier when multiple I/O processors creates multiple data files.

Finally, Panda supports many other useful features such as adaptive mesh refinement (AMR) and automatic data migration. For the **IOPanda** thorn, these features will be included soon.

3 Thorn IOPanda

Thorn **IOPanda** registers an I/O method called `IOPanda` with the I/O interface in the flesh. This method creates three-dimensional output of 3D grid functions. (and possibly multidimensional arrays, which are not implemented fully in Cactus yet). Panda library integrated into **IOPanda** thorn has been modified and currently it supports to write array data in IEEEIO binary format. It will be modified again so that it can support output in HDF5.

Data written by panda goes into files named `"<varname>.<rank_of_io_processor>"` and will be placed in the directory which is specified in the input paramemter file. Futher processing of these data can be done with visualization tools like Amira, AVS, and IDL.

Checkpoint/recovery operations are not implemented yet and will be included in the thorn soon.

4 Comments

Since **IOPanda** uses parameters from **IOUtil** it also needs this I/O skeleton thorn compiled into Cactus and activated.

When using **IOPanda** thorn, sometimes an error regarding MPI_TYPE_MAX can occur. If this happens, simply increase the value using `setenv MPI_TYPE_MAX <larger number than current value>`.

If arrays cannot be divided evenly by the mesh determined by Cactus (i.e. dividing 10x10 array into nine chunks using 3x3 mesh), **IOPanda** can't handle this case because it divides array in a way different from Cactus. Please avoid this case for now.