pRPL 2.0: a parallel Raster Processing Library supporting dynamic data load-balancing

Qingfeng (Gene) Guan
gguan2@unl.edu
Center for Advanced Land Management Information Technologies
School of Natural Resources
University of Nebraska – Lincoln

Parallel computing, in contrast to sequential computing, is the use of multiple processing units (computers, processors, and processes) working together on a common task in a concurrent manner in order to achieve higher performance, usually measured by computing time. For geospatial practitioners, parallel computing provides a means to overcome computational performance limits to solve geospatial problems in a shorter period of time, even to solve problems that are computationally infeasible using sequential computing technology.

The raster data model, one of the most widely used data models for geospatial data, represents phenomena in geographical space as a grid of cells with attribute values. From a parallel computing perspective, raster is born to be parallelized. A raster dataset is essentially a matrix of values, each of which represents the attribute of the corresponding cell of the field. A matrix can be easily partitioned into sub-matrices and assigned onto multiple processors so that the sub-matrices can be processed simultaneously and a higher processing speed will be reached.

A parallel Raster Processing Programming Library (pRPL), a C++ programming library, was developed by the author to encapsulate complex parallel computing utilities and routines specifically for raster processing (e.g., raster data decomposition, distribution and gathering among multiple processors, inter-processor communication and data exchange), and enable users to easily parallelize almost any raster-processing algorithm with any arbitrary neighborhood configuration. From a software architecture perspective, pRPL serves as a middleware connecting the general-purpose parallel programming library (i.e., Message Passing Interface, or MPI) and the application-specific raster-processing programs. It provides GIScientists an easy-to-use toolkit to exploit the great computing power of high-performance computing facilities, i.e., providing transparent parallelism, and more importantly, a test-bed for computationally intensive geospatial models and a problem-solving environment for previously computationally infeasible approaches.

pRPL is a general-purpose programming library that provides generic support for raster processing, including local-scope, neighborhood-scope, regional-scope, and global-scope algorithms as long as they are parallelizable. pRPL supports multi-layer algorithms that are commonly used in geospatial applications. pRPL provides multiple data-decomposition methods for users, including a spatially-adaptive quad-tree-based (QTB) decomposition method for cases when the computational intensity is extremely heterogeneous over space. The “Update-on-Change” and Value-headed Global-index Stream techniques developed for pRPL help to reduce the communication overhead for data exchange among
the processors, hence reduce the computing time. Furthermore, the “edgesFirst” and non-blocking communication techniques overlap the computation and communication, which also helps reduce the computing time. pRPL organizes processors into groups, and supports data-task hybrid parallelism which is innovative for parallel raster processing and especially useful when handling massive-volume datasets and a large number of parallelizable tasks at the same time. With grouped processors, dynamic load-balancing among tasks can be implemented with ease. pRPL also provides an intuitive programming guideline for users to implement application-specific algorithms, and requires minimal parallel programming knowledge.

A parallel geographic Cellular Automata model, pSLEUTH, was developed using pRPL v1.0. pSLEUTH fully utilizes the features of pRPL, parallelizing the SLEUTH model. Experiments with real-world datasets showed that pSLEUTH greatly reduced the computing time needed for the calibration process, and yields fairly high performances, achieving a speed-up of 24 using 32 processors.

However, the version 1.0 pRPL only supports static load-balancing for data parallelism, and requires the master node to read the whole raster data into the memory before the decomposition and distribution processes. While dealing with extremely large raster datasets, pRPL v1.0 can hardly perform efficiently, because the lacking of support for dynamic data load-balancing causes severe degradation of performance. pRPL v2.0 is being developed to overcome the problem and improve the performance by implementing the support for dynamic data load-balancing. As shown in Figure 1, the output grid is decomposed into equal-area sub-grids, and the information of the output sub-grids is used to appropriately decompose the input grid (often into unequal-area sub-grids). A master-worker-based dynamic data load-balancing technique is then deployed. The master node holds a data-farm, dynamically reads input sub-grids into the memory, assigns and distributes them to worker nodes in response to their requests. The worker nodes, on the other hand, request and receive tasks, apply the raster-processing algorithm on the local sub-grids, and send them back to the master for output.

![Figure 1 Dynamic data balancing in pRPL v2.0](image)

pRPL v2.0 will be used to develop a rapid raster projection transformation system called pRasterProject. In particular, the algorithm of MapImage, a software package developed by the USGS for raster projection transformation, will be parallelized using pRPL.