Wavelet Subband Coding of Computer Simulation Output
Using the A++ Array Class Library

Jonathan N. Bradley, Christopher M. Buislawn, Daniel J. Quinlan, and
Hua D. Zhang
Los Alamos National Laboratory, Los Alamos, NM 87545

Veyis Nuri
School of EECS, Washington State University, Pullman, WA 99164

Abstract

This work focuses on developing discrete wavelet transform/scalar quantization data compression software that can be ported easily between different hardware environments. This is an extremely important consideration given the great profusion of different high-performance computing architectures available, the high cost associated with learning how to map algorithms effectively onto a new architecture, and the rapid rate of evolution in the world of high-performance computing. Our approach is to use the A++/P++ array class library, a C++ software library originally designed for adaptive mesh PDE algorithms. Using a C++ class library has the advantage of allowing us to write the scientific algorithm in a high-level, platform-independent syntax; the machine-dependent optimization is hidden in low-level definitions of the library objects. Thus, the high-level code can be ported between different architectures with no rewriting of source code once the machine-dependent layers have been compiled. In particular, while “A++” refers to a serial library, the same source code can be linked to “P++” libraries, which contain platform-dependent parallelized code.

The present paper compares the overhead incurred in using A++ library operations with a serial implementation (written in C) when compressing output of a global ocean circulation model currently running at the Los Alamos Advanced Computing Lab. We compare performance on two vastly different platforms: a Sun SparcStation 2 and a Cray Y-MP/M98. Because the multiprocessor (P++) array class library is still being written for the Cray, all testing was conducted on a single Y-MP processor using the A++ library; the same A++ library source code was used on both machines. When the P++ library is completed, we will be able to make the same comparison of serial implementation with a parallelized P++ implementation by linking the same application code to a P++ library.

Testing presently reveals a performance penalty suffered in exchange for the platform-independent high-level A++ syntax. On the Sparc 2, timings indicate that the overhead penalty is not made up in the large-array limit. Performance comparisons on the Cray are more encouraging but are complicated by the way in which different portions of the code vectorize. In particular, timings indicate that the A++ DWT is vectorizing much more efficiently than the serial C implementation; results on scalar quantization vectorization are less positive at present. In a parallel environment using the same source code with a P++ library, we expect that the overhead should be modest when compared to the difficulties in parallelizing the serial C code.