Tech Notes

Parallel Vector Tile Optimizing Library

A C++ library that allows cross-platform software portability without sacrificing high performance

Researchers at MIT Lincoln Laboratory developed the Parallel Vector Tile Optimizing Library (PVTOL) to address a primary challenge faced by developers of embedded signal processing applications: how to write programs at a high level while still achieving performance and preserving the portability of the code across platforms. PVTOL provides a layered architecture that functions as a middleware to isolate application code from hardware code, thus avoiding the need for computation and communication libraries customized to a specific hardware.

The Need for Software Portability

Real-time signal processing consumes most of the world's computing power. Wireless communication, radar signal processing, medical imaging, and scientific data processing are just some of the applications dependent upon the immediate translation of signals into useful "products," e.g., a phone call or an on-screen visualization of an aircraft's flight path. As the capabilities of applications grow, their large computation and communication requirements can only be met with the use of multiple processors and fast networks.

Software that manages the large number of processors and complex communication networks while still achieving high performance is a major concern for application and system developers. Moreover, the high cost of creating advanced code for demanding multicore or parallel processing systems has driven the move from low-level, machine-specific software to high-level, portable software. A key technical hurdle is balancing the demands of application performance, software portability, and developer productivity.

For developers working on military systems (Figure 1), the hurdle is greater. Military sensing platforms often use a variety of signal processing systems at different stages of a mission. The software for sophisticated Department of Defense (DoD) radar, sonar, and imaging sensors requires complex algorithms implemented on complex hardware. In addition, the substantial investment required to develop unique DoD systems compels long system lifecycles—a demand that means software must be highly portable to stay abreast of rapidly changing hardware.

Figure 1. Military platforms are among the largest and most complex users of real-time signal processing applications. Performance demands for these applications are exceptionally high, and portability is very important because of the long life cycles of the systems.

Technical Point of Contact
Edward M. Rutledge
Embedded and High Performance Computing Group
rutledge@ll.mit.edu
781-981-0274

For further information, contact
Communications and Community Outreach Office
MIT Lincoln Laboratory
244 Wood Street
Lexington, MA 02420-9108
781-981-4204

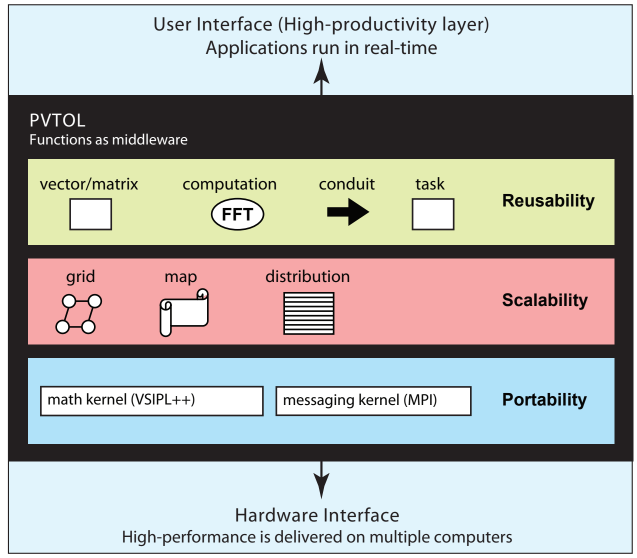

Figure 2. The layered architecture of PVTOL acts as a middleware between the application (user interface) and the hardware code.

Solution: Innovative Middleware

The traditional application-development approach that requires customized libraries leads to software that is complex to write, does not run on multiple platforms, and is dependent on the number of processors on which the application is deployed. Although this approach achieves high performance, it is not cost-effective because programmer effort is high and software reusability is low.

Built upon concepts behind the Laboratory's earlier Parallel Vector Library, PVTOL allows signal processing algorithm code to be written at a high level by using task and data *objects* (e.g., variables, matrices, vectors) that may be parallel but that insulate the application from details of how those objects are mapped to the parallel architecture or how they synchronize and communicate. PVTOL programs scale easily to changes in the maps of the objects. Mapping is specified separately from the algorithm code, so the algorithm code does not have to be changed to take advantage of a larger or smaller number of processing elements. Also, because the PVTOL application programming interface is constant across platforms, the application code does not have to be changed when the application is ported to a new hardware platform.

PVTOL increases portability across platforms, resulting in increased programmer productivity and sustained program performance.

Figure 2 illustrates PVTOL's role in an application. First, the application is written using the reusability layer. Objects such as matrices, vectors, and computations are provided and serve as the primary means for specifying the signal or image processing functionality. Tasks and intertask communication objects, called conduits, are provided to allow the system to be broken into modules and to allow for the queuing of objects and round-robin scheduling of tasks.

Second, at the heart of PVTOL is the scalability layer. Each PVTOL component in the reusability layer has a map from the scalability layer that is read in at initialization from a file. The map specifies how the object is distributed over the parallel machine. Changing the scale of an application is accomplished by changing the maps. Because these are read in from a file, only the file content needs to be changed. The signal or image processing application source code remains untouched. This decoupling of the mapping process from the functionality is key to providing a scalable system.

At the portability layer, PVTOL uses hardware-tuned standards: the parallel C++ binding of the Vector Signal Image Processing Library (VSIPL++) and the message passing interface (MPI). The use of these standards ensures wide portability, and PVTOL has been successfully implemented on a wide range of workstation, cluster, and embedded architectures.

Advantages of PVTOL

Lincoln Laboratory's PVTOL technology addresses primary concerns of application developers:

- *Productivity:* By encapsulating interprocessor communication details and low-level computation details within high-level parallel objects, PVTOL allows developers to work at a high level of abstraction. Also, because developers do not have to rewrite application code to adapt to new hardware or to add features, they realize improved efficiency.
- *Portability:* By providing a high-level application programming interface that is consistent across hardware platforms and by decoupling an application's functionality from its hardware mapping, PVTOL increases portability across platforms, resulting in not only increased programmer productivity but also sustained program performance despite continual hardware upgrades. The PVTOL approach can preserve more than 90% of signal processing application code that is ported to a new hardware architecture.
- *Performance:* By using advanced optimization techniques and industry standards in hardware code, PVTOL achieves almost the same high performance as achieved by approaches utilizing hardware-specific code and custom libraries.

Future

Lincoln Laboratory expects that concepts demonstrated in PVTOL will influence the VSIPL++ standard, into which the Laboratory's PVL concepts have already been incorporated. Extensions to VSIPL++ to make it more compatible with task-parallel frameworks such as PVTOL have been proposed and prototyped, and are being evaluated by the high-performance computing community. ■

Additional Reading

J. Lebak, J. Kepner, H. Hoffmann, and E. Rutledge, "Parallel VSIPL++: An Open Standard Software Library for High-Performance Parallel Signal Processing," *Proceedings of the IEEE,* vol. 93, no. 2, 2005.

J. Kepner and J. Lebak, "Software Technologies for High-Performance Parallel Signal Processing," *Lincoln Laboratory Journal,* vol. 14, no. 2, 2003.

Opinions, interpretations, and recommendations herein are not necessarily endorsed by MIT Lincoln Laboratory's government sponsors. Work described in this document is performed under the prime contract with the U.S. Air Force, FA8721-05-C-0002.