Applying Advanced Computing to Improve High-Fidelity Radar Data Simulations

Christopher Hulbert (cch@isl-inc.com), Jameson Bergin (jsb@isl-inc.com),
Paul Techau (pmt@isl-inc.com)
Information Systems Laboratories, Inc.

Abstract

High-fidelity radar (IQ) data simulations involve many complex operations, especially in clutter simulations. This paper describes the parallelization of Matlab simulation software using Matlab MPI, its transition to compiled language functions, and the parallelization and optimization of supporting libraries. Because of the structure of the simulation, parallel computation architectures can reduce simulation time from weeks and days to hours and in some cases minutes. Timing results using the AFRL HHPC system are presented.

1. Introduction

High-fidelity radar simulations have proven useful for analyzing the performance of adaptive radar systems over relatively short processing intervals on the order of a few minutes [e.g. 1,2]. Characterizing system performance over longer time periods (e.g., tracking performance), however, requires significantly more simulated data which can often take weeks to months to generate on single node computers. This is not an acceptable time-frame for efficient analysis and often severely limits the use of high-fidelity simulations to completely characterize the performance of a given radar system. Unfortunately, existing clutter simulations must sacrifice fidelity to address these longer run times. An alternative to reducing fidelity is to apply high performance computing to speed up the high-fidelity simulations.

The objective of the radar clutter data simulations is to model the site-specific clutter for a given radar geometry. A range swath and its ambiguous range swaths can be divided into a number of clutter patches of any size. Using these discrete patches we can define a typical clutter signal model as [3],

$$\mathbf{x}_c(t) = \sum_{p=1}^{P} \alpha_p \mathbf{s}_p(t) \circ \mathbf{v}_p \circ \mathbf{u}_p \in C^{NM \times 1},$$

where α_p and $\mathbf{v}_p$ are the complex scattering amplitude and space-time steering vector for the p^{th} clutter patch, $\mathbf{s}_p(t)$ is a vector that accounts for range walk and bandwidth effects for an arbitrary waveform, and $\mathbf{u}_p$ is a unit energy modulation processes that accounts for temporal and spatial modulation mechanisms respectively. The total number of simulated clutter patches is P and the number of receiver channels and pulses is N and M, respectively. The operator $\circ$ is the Hadamard element-wise vector product.

The equation shows that each term is dependent only on the parameters of the current clutter patch, and thus allows for easy parallel implementation. This also illustrates that the simulation time is characterized by two main components, the number of clutter patches and computations for each patch.

ISL has developed the Matlab SCATS and Radar Simulation toolboxes for this application. Matlab SCATS is a toolbox for RF signal environment characterization including: power, Doppler, azimuth angles, elevation angles, propagation factors, etc. for each scatterer in a scenario. The radar simulation toolbox is a general simulation framework that takes user inputs, analyzes the RF signal environment using Matlab SCATS, and then generates the IQ data. All initialization and core functions are called using Matlab function handles. This design gives the user complete control over the simulation and allows easy development and implementation of new algorithms.

These toolboxes were used extensively under the recent Defense Advanced Research Projects Agency's (DARPA) KASSPER program [4] to generate high-fidelity radar data to support advanced signal processing algorithm development.

2. Software Development

ISL's software development goals are to create a portable and efficient implementation of a radar simulation tool. Portability is important as various platforms and architectures are tested. The efficient implementation has several components including parallel implementation and optimization of libraries used. An added benefit of both a Matlab

and C implementation is data quality assurance (verifying both sets of code give the same result).

2. 1 Parallel Matlab Prototype

The radar simulation toolbox uses MatlabMPI to distribute the cells for each simulation. When a simulation is complete, the IQ data is combined using a sum reduction. Further performance improvements per node in Matlab were desired to improve simulation time. A desirable approach was to begin rewriting many of these tools in a compiled language and use Matlab's External interface (MEX) functions as this allows continued development in the Matlab environment while optimizing lower-level functions. Fortran and C/C++ were considered, but because of the cleaner interface from Matlab to C and ISL's experience with C, C was chosen as the target language. To aid in portable code and easier implementation of vectorized operations, the Vector, Signal, and Image Processing Library (http://www.vsipl.org) was used. The VSIPL API standard is well supported by many hardware and software vendors. HPEC-SI has also picked up the program implementing a C++ and parallel C++ standard known as vsipl++ and parallel vsipl++. Using the VSIPL library as a computation base has improved performance 2-10 times.

The VISPL library initially used was the reference library distributed on the VSIPL website. Since our current simulation architectures include consumer processors such as Intel Pentium/Xeon and AMD Athlon64/Opteron processors, it was desired to improve the VSIPL optimization on these architectures. Optimizations made to the VSIPL library include using the AMD/Intel Math libraries as well as introducing OpenMP directives for threading.

2.2 MPI Parallel Implementation

In the Matlab-independent software, MPI is used. Implementations on ISL's cluster use mpich2 – a freely available MPI-2 standard implementation from Argonne National Laboratory. On the AFRL cluster, mpich1 was used. Additional tests will likely also use the AFRL Myrinet connection.

3. Performance Results

Under a contract with the Air Force Research Laboratory (AFRL) Rome Research Site (RRS), ISL has been given access to their high-performance computing resources. The main system used thus far is the Heterogeneous High Performance Computer which is a 48 node dual Intel Xeon 2.2GHz with 4 Gb of SDRAM per node. Each node is connected on both a Gigabit Ethernet and Myrinet 2000 backbone. When writing the C code for the MEX functions, the code that actually performed the same functions as the Matlab code were separated out and placed in libraries. This design was employed as it allows an easier transition to software independent of Matlab.

Simulation times for 1, 5, 10, 20, 30, 40, and 60 nodes are given in figure 1. The blue curve is ideal linear speedup T_1/T_p where T_i indicates the expected simulation time using i processors. The red circle markers are actual performance times.

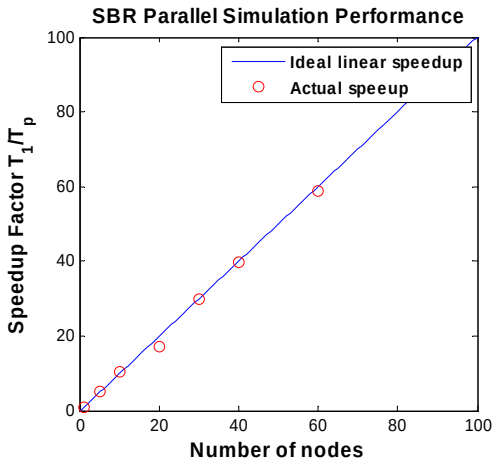

Figure 1 HHPC timing results

4. Summary and Future Work

Future optimization work is an open area of consideration. Some areas being considered include Lincoln Laboratories' PVL, VSIPL++ parallel, FPGA's, and GPU clusters. ISL has used PVL in other software and the AFRL HHPC cluster has a Wildstar II FPGA board on each node, so these are likely next steps. Additionally, ISL will continue to improve on the VSIPL performance.

References

[1] P. M. Techau, *et. al.,*"Performance bounds for hot and cold clutter mitigation," *IEEE Trans. on AES*, vol. 35, pp. 1253-1265, October, 1999.

[2] D. Page, *et al.*, "Improving knowledge-aided STAP performance using past CPI data", *Proceedings of the IEEE 2004 Radar Conference*, April 26-29, 2004, pp. 295 – 300.

[3] P. M. Techau. "A general clutter signal model". Draft Technical Note, Information Systems Laboratories, Inc. November 2005.

[4] www.darpa.mil/SPO/programs/kassper.htm