Adaptive Computing in NASA Multi-Spectral Image Processing

Mark L. Chang, Scott A. Hauck
Department of Electrical and Computer Engineering
Northwestern University
Evanston, IL 60208 USA
{mchang, hauck}@ece.nwu.edu

Introduction

With the anticipated launch of the first Earth Observation System satellite, the AM-1 (also known as Terra [Figueiredo99]), ground processing systems will need to process more data than ever before. Much more, in fact, as Terra will produce more data in its first six months of operation than NASA has collected since it's inception [Graessle98]. To be able to handle the processing loads that are anticipated, NASA plans to utilize high-performance parallel-processing systems in Distributed Active Archive Centers (DAACs) spread around the country.

Our work, combined with the efforts of others, strives to complement the DAACs using workstations equipped with an adaptive compute engine, such as an assortment of field-programmable gate arrays (FPGAs). These adaptive compute engines allow engineers to implement their algorithms, or portions of their algorithms, in a hardware substrate with hardware-like speeds, while maintaining the flexibility of software. Furthermore, adaptive computing engines, with their reconfigurable nature, can often be reprogrammed in a matter of milliseconds, allowing a single piece of silicon to be reused for multiple tasks.

Multi-Spectral Image Classification

In our work, we are focusing on accelerating a typical multi-spectral image classification algorithm. This algorithm uses multiple spectrums of instrument observation data to classify each pixel into one of many classes. In our implementation, these classes consist of terrain types, such as urban, agricultural, rangeland, and barren, although they could be any significant distinguishing attributes. One proposed scheme to perform this automatic classification is the Probabilistic Neural Network classifier as described in [Chettri92]. In this scheme, each multi-spectral image pixel vector is compared to a set of "training pixels" that are known to be representative of a particular class. The probability that the pixel under test belongs to the class under consideration is given by the formula:

$$f(\vec{X} \mid S_k) = \frac{1}{(2p)^{d/2} s_k^d P_k} \sum_{i=1}^{P_k} \exp\left(-\frac{1}{2s_k^2} \cdot (\vec{X} - \vec{W}_{ki})^T (\vec{X} - \vec{W}_{ki}) \right)$$

$\vec{X}$: pixel vector

$\vec{W}_{ki}$: Weight i of class k

d : number of bands

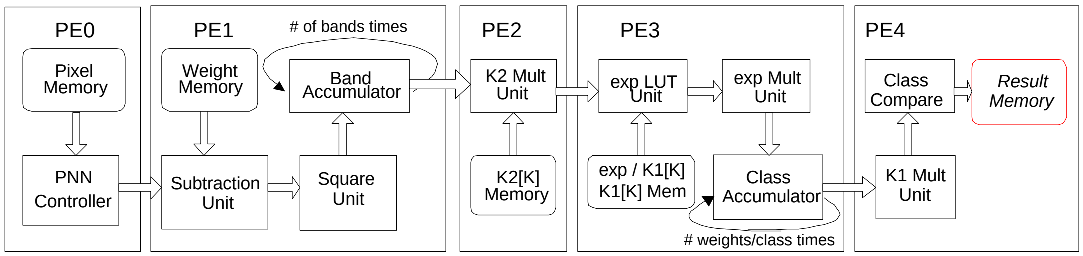
k : number of classes

P_k : number of weights per class

The highest value that this probability formula obtains for a class *k* indicates that the pixel vector matches most closely with that class.

Implementation

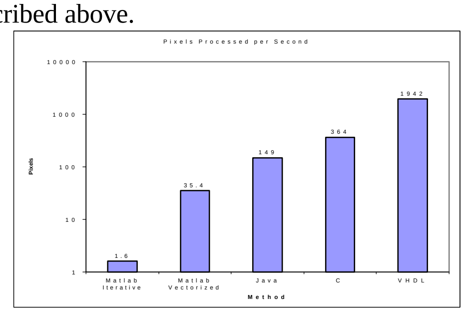
The PNN algorithm was implemented in an array of languages as well as on a COTS adaptive computing engine. The languages included Matlab, Java, and C, while the hardware version was written in VHDL. The hardware used was the Annapolis Microsystems WildChild FPGA co-processor. This board contains an array of Xilinx FPGAs–one "master" Xilinx 4028EX and eight Xilinx 4010E's, referred to as Processing Elements (PEs)–interconnected via a 36-bit crossbar and a 36-bit systolic array. Each PE(0) has 1MB of SRAM while each PE(1-8) has 512K of SRAM. The board is installed into a VME cage along with its host, a FORCE SPARC 5 VME card. The hand-mapping of the algorithm is shown below. (*K1* and *K2* are merely the constants inside the exponentiation and outside the sum.)



Results

The Matlab code was implemented in two ways. First, a simple iterative approach was used which, for those who have used Matlab, is always the slowest implementation. The second implementation took advantage of Matlab's vectorized operations and obtained much better performance. The results are shown below in terms of pixels processed per second. All software versions were timed on an HP C180 workstation, while hardware timings were done on the system described above.

	Time (sec)	Pixels/sec	Lines of code
Matlab Iterative	162000	1.6	39
Matlab Vectorized	7200	35.4	27
Java	1755	149	474
C	720	364	371
VHDL	135	1942	2205



It is noteworthy that this implementation utilizes only half of the available PEs on the WildChild board. Since the evaluation of each pixel is independent in the PNN algorithm, we should obtain linear speedups by using the remaining four FPGAs available. This minor modification is currently being implemented.

The MATCH Project

We have found that the scientists that would benefit the most from adaptive computing technologies are unwilling to master the complexities that are associated with the development of adaptive-computing-enabled applications. The complexity in design can be attributed to the languages involved and the current state of tools that aid in simulation, debugging, and implementation. Even with these tools, targeting these adaptive systems is still largely a manual task that requires much hardware expertise.

At Northwestern, we are working on the MATCH project [Banerjee99]. The focus of this project is the development of a compiler that endeavors to take Matlab codes and automatically translate them into FPGA- and DSP-based implementations. This will allow scientists to achieve high performance without extensive hardware expertise.

Our work with the multi-spectral image classifier and other NASA applications will serve as a driver application for the development of the MATCH compiler. This work will lead to the creation of module libraries and compilation support necessary to support NASA applications. Additionally, we envision the development of optimization tools to aid the designer in hand-optimizing the code generated by the MATCH compiler. Optimizations such as bit-width tuning and critical path acceleration must be addressed to obtain the highest performance from the target system.

References

[Figueiredo99] Figueiredo M. A., Flatley T. P., Stakem P. H., Hines T. M; "Extending NASA's Data Processing to Spacecraft", IEEE Computer Volume 32 number 6, pages 115-118, June 1999.

[Chettri92] S. R. Chettri, R. F. Cromp, M. Birmingham; "Design of neural networks for classification of remotely sensed imagery", Telematics and Informatics, Vol. 9, No. 3, pp. 145-156, 1992.

[Graessle98] Graessle T., Figueiredo M.; "Application of Adaptive Computing in Satellite Telemetry Processing"; 34th International Telemetering Conference - San Diego, California, October 26-29, 1998.

[Banerjee99] P. Banerjee, A. Choudhary, S. Hauck, N. Shenoy; "The MATCH Project Homepage", *http://www.ece.nwu.edu/cpdc/Match/Match.html*