Applications Kernels on Graphics Processing Units

An analysis of Hidden Markov Models, Support Vector Machines, Hypersectral Imaging, and Latent Semantic Indexing

Sean Ahern, (seanahern@llnl.gov), David Bremer, (dbremer@llnl.gov),
John R. Johnson, (jjohnson@llnl.gov), Holger Jones, (holgerjones@llnl.gov), Yang Liu,
(liu24@llnl.gov), Jeremy Meredith, (j-meredith@llnl.gov),Sheila Vaidya, (vaidya1@llnl.gov)
Lawrence Livermore National Laboratory

Candace Culhane
Department of Defense

1. Introduction

The GAIA project at LLNL has been actively investigating the efficacy of data-parallel computation on commodity graphics processing units (GPUs). Our initial thrust has been centered on algorithms found in the image processing domain such as convolution, correlation, FFT, and matrix multiply. For the past two years the GPGPU community has seen a relatively large proliferation of algorithms drawn from a wide variety of application domains which have been mapped onto the GPU. We will describe GPUs performance results from extending the application space to application kernels that are used in *protein matching and speech processing* (HMMs), *seismic and plume modeling* (Support Vector Machines), *hyperspectral imagery*, and *text processing* (Latent Semantic Indexing).

2. GPU numerical precision

The processing pipeline of a typical GPU such as the NV40/NV45 is currently fixed at 32bit single precision, so many practitioners are skeptical when approaching the effort of porting their algorithms. Some preliminary efforts [1,2] to reverse engineer the floating point behavior of GPUs show that there are significant issues. GPUBench takes the tactic of comparing the intrinsic operations (mul/div, sqrt, etc) and compares the precision with a CPU double, while the other uses a subset of the "Berkeley FP Paranoia" benchmarking suite [3] to expose the GPU behavior. We elaborate on this discussion by illustrating error propagation for single precision FFT and accuracy validation by solving the heat conduction equation. Elsewhere we have shown that nearly double precision arithmetic can be performed in software on GPUs with only minimal slowdown, [8, 9].

3. Application Kernels on GPUs

3.1 **HMMs**

We introduce the implementation of Viterbi HMM algorithms used for protein sequence matching [4], and speech processing. These efforts leverage the work of the open-source implementation HMMer for protein databases, and we elaborate to more general HMMs in the context of speech processing as found for example in the MS. State ISIP Package [5]. ClawHMMER a streaming implementation of hmmsearch has been shown to perform on average about twice as fast as a heavily optimized PowerPC G5 (2.5 GHz) and seventeen times as fast as the standard Intel P4 (2.8 GHz)

3.2 Support Vector Machines:

SVMs are trained to recognize and categorize features in very large data sets, and the training data sets are generated to differentiate between categories using physically realistic forward models. [6, 7]

The goal is to find the parameters of a Support Vector Machine for regression such that

$m = SVM(d)$

The SVM will be of the form

$$m = \sum_{i=1}^{n} a_i K(d_i, d)$$

where:

m = model(row vector)

d = data(row vector)

a = SVM coefficients (the goal is to find these)

n = number of training examples

K = kernel function or kernel subroutine

We use the GPU to calculate the Gram matrix K(*,*) of training data utilizing a Gaussian kernel; this usually considered the most computationally intensive step and we are already seeing speedups of factors of 10 or more.

3.3 Hyperspectral Imagery

This section explores computation in the context of hyperspectral imaging, (HSI) applications [7]. The computation of the covariance matrix is the most computationally intensive component of an HSI application so any speedup of this computation will significantly improve the full HSI application performance. We have preliminary results showing that there is roughly an 8x speedup of the computation of the covariance matrix on a GPU.

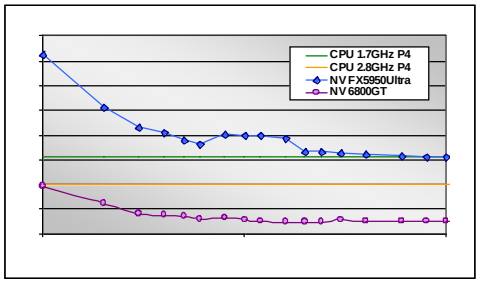

Figure 1. Performance Comparison of Covariance Calculation on CPU and GPU

3.4 Latent Semantic Indexing

LSI is used to support semantic retrieval of large text corpora. LSI algorithms revolve around using the Singular Value Decomposition to produce statistically uncorrelated index variables. In this section we discuss using the GPU to solve the SVD using block-Jacobi technique [6]. Current results show performance improvements on the order of five times that of Matlab.

GPGPU developments

We will describe the recent technology advancements in general purpose computing for GPUs such as the use of the OpenGL Frame Buffer Extension, GPU technology roadmaps, multi-GPU capable workstations, and some efforts towards tandem CPU-GPU multiprocessing. We will also discuss limitations and challenges associated with GPUs. GPU applications are often memory-bound, and the memory fetch latencies can be concealed by co-issuing arithmetic instructions. However, the speed of arithmetic instructions will increase faster than bandwidth in future iterations of graphics cards, so it may be more difficult to conceal these latencies.

References

[1] I. P. Hanrahan, J. Sugarman, K. Fatahalian, M. Houston, T. Foloy, D. Horn: http://graphics.stanford.edu/projects/gpubench/.

[2] K. Hillesland and A. Lastra. “GPU floating-point paranoia.” GP2, 2004.

[3] Karpinski, R., "Paranoia: a Floating-Point Benchmark", Byte, February 1985.

[4] D. Horn, M. Houston, “ClawHMMER: A Streaming HMMer-Search Implementation”, Submited to SC05

[5] http://www.isip.msstate.edu/projects/speech/

[6] H. Kuzma, J. Rector, “Contaminant Plume Analysis using Support Vector Machines, UCB Progress Report to LLNL.

[7] D. Bremer, J. Johnson, H. Jones, et. al. “Graphics Processing Units (GPUs) for General Purpose High Performance Computing”, submitted to SC05

[8] J. Meredith, D. Bremer, et. al., “The GAIA Project: Evaluation of GPU-Based Programming Environments for Knowledge Discovery,” HPEC 2004, Boston.

[9] D. Bremer, J. Johnson, et. al., “Precision and Accuracy on Graphics Processing Units (GPUs),” submitted to SC05.