New Data Structures for the Cell Processor

Fred Gustavson – Jerzy Wasniewski

IBM Thomas J. Watson Research Center – Informatics and Mathematical Modelling, DTU

This Work Shop has Seven Parts

Part 1: Blocking and New Generalized Data Structures (NDS)

The aim is to obtain very high performance dense linear algebra algorithms on Cell. "Cache Blocking"is still key. The Algorithms and Architecture Approach leads to the use of NDS and kernels for DLA (Dense Linear Algebra) Factorization algorithms. It will be shown that all DLAFA (Dense Linear Algebra Factorization Algorithms) are almost all matrix multiply algorithms. DGEMM (General Matrix Matrix Multiplication) is flawed as its API uses standard CM (Column Major) / RM (Row Major) array formats of Fortran and C. NDS overcomes the flaw. A paper of Jack Dongarra's team states that use of NDS is the key factor for getting performance on Cell for the Linpack Benchmark.

Part 2: One, two, three & Higher Dimensions

Matrices are two dimensional objects. Fortran and C store matrices in one dimensional arrays. The main theorem of Dimension Theory proved by LEJ Brouwer in 1913 states that representing a 2-D matrix as a 1-D object will not preserve closeness of sub-matrix elements. NDS does this.

Part 3: Block In-place Transpose of a Rectangular Matrix

In-place transposition was an active research area in the late 1950's to the middle 1970's. Gustavson began to study this problem anew in relation to New Data Structures (NDS) for Dense Linear Algebra (DLA). The new results give fast algorithms for in-place transposition in particular as well as in-place data transformations in general. These new fast algorithms incorporate "cache blocking"and NDS as their major high performance components.

Part 4: Basic Cell Architecture

The Cell architecture grew from a challenge posed by Sony and Toshiba to provide power-efficient and cost-effective high-performance processing for a wide range of applications, including the most demanding consumer appliance: game consoles. Cell - also known as the Cell Broadband Engine Architecture (CBEA) - is an innovative solution whose design was based on the analysis of a broad range of workloads in areas such as cryptography, graphics transform and lighting, physics, fast-Fourier transforms (FFT), matrix operations and scientific workloads.

Cell is a heterogeneous chip multiprocessor that consists of an IBM 64-bit Power Architecture core, augmented with eight specialized co-processors based on a novel single-instruction multiple-data (SIMD) architecture called Synergistic Processor Unit (SPU), which is for data-intensive processing, like that found in cryptography, media and scientific applications. The system is integrated by a coherent on-chip bus.

Part 5: Matrix Multiply On Cell

Based on Part 1, we need to consider matrix multiply on Cell if we want to be able to do Dense Linear Algebra Factorization Algorithms on Cell. A novel of feature of Cell allows one to overlap "communication with computation". An old result of Agarwal, Gustavson and Zubair in 1993 showed that this feature always led to perfect speed-up for distributed memory computing matrix multiplication. Thus, it turns out that matrix mutiplication on Cell can be done at the peak GFlop rate of Cell.

Part 6: A Cholesky Factorization on Cell

This lecture reviews a paper on Cholesky Factorization on Cell by Kurzak, Buttari and Dongarra (Lapack Working Note 184). Also, improvements and relationships between this paper and Gustavson's research with Umeå and UNI•C / IMM over the last ten years is discussed.

Part 7: Three versions of High Performance Minimal Storage Cholesky Algorithm which uses New Data Structures: Recursion, Rectangular Full Packed (RFP), and Block Packed Hybrid Formats

We describe new data formats for storing triangular, symmetric, and Hermitian matrices. The standard two dimensional arrays of Fortran and C (also known as full format) that are used to store triangular, symmetric, and Hermitian matrices waste nearly half the storage space but provide high performance via the use of level 3 BLAS. Standard packed format arrays fully utilize storage (array space) but provide low performance as there are no level 3 packed BLAS.

We combine the good features of packed and full storage using the new formats to obtain high performance using (level 3) BLAS. Also, these new formats require exactly the same minimal storage as LAPACK packed format. These new formats even outperform the LAPACK full format for some computer platforms. The Block Packed Hybrid Format works well for multi-core processors.