High Performance Linear Algebra Algorithms: 
An Introduction

Organizers: Fred G. Gustavson\(^1\) and Jerzy Waśniewski\(^2\)

\(^1\) IBM T.J. Watson Research Center  
Yorktown Heights NY 10598, USA  
fg2@us.ibm.com

\(^2\) Department of Informatics & Mathematical Modeling  
of the Technical University of Denmark  
DK-2800 Lyngby, Denmark  
jw@imm.dtu.dk

1 Introduction

This Mini-Symposium consisted of two back to back sessions, each consisting of five presentations, held on the afternoon of Monday, June 21, 2004. A major theme of both sessions was novel data structures for the matrices of dense linear algebra, DLA. Talks one to four of session one all centered on new data layouts of matrices. Cholesky factorization was considered in the first three talks and a contrast was made between a square block hybrid format, a recursive packed format and the two standard data structures of DLA, full and packed format. In both talks one and two, the first two new formats led to level three high performance implementations of Cholesky factorization while using exactly the same amount of storage that standard packed format required. Of course, full format requires twice the amount of storage of the other three formats. In talk one, John Reid presented a definitive study of Cholesky factorization using a standard block based iterative Cholesky factorization, \(^1\). This factorization is typical of Lapack type factorizations; the major difference of \(^1\) is the type of data structure it uses: talk one uses square blocks of order \(NB\) to represent a lower (upper) triangular matrix. In talk two, Jerzy Waśniewski presented the recursive packed format and its related Cholesky factorization algorithm, \(^2\). This novel format gave especially good Cholesky performance for very large matrices. In talk three, Jerzy Waśniewski demonstrated a detailed tuning strategy for talk one and presented performance results on six important platforms, Alpha, IBM, Intel, Itanium, SGI and Sun. The performance runs covered the algorithms of talks one and two as well as Lapack’s full and packed Cholesky codes, \(^3\). Overall, the square block hybrid method was best but was not a clear winner. The recursive method suffered because it did not combine blocking with recursion, \(^4\). Talk four, presented by Fred Gustavson, had a different flavor. Another novel data format was described which showed that two standard full format arrays could represent a triangular or symmetric matrix using only a total storage that was equal to the storage of standard packed storage, \(^5\). Therefore, this format has the same desirable property of standard full format arrays: one can use standard level 3 BLAS, \(^6\) as well as some 125 or so full format Lapack symmetric / triangular routines on it. Thus new codes written for the new format are trivial to produce as they mostly consist of just calls to already existing codes. The last

---

© Springer-Verlag Berlin Heidelberg 2006
talk of session one, by James Sexton was on the massively parallel IBM BG/L machine consisting of up to 65,536 processors. Jim presented some early Linpack benchmark numbers which were quite impressive, [7]. A surprising property of this machine was its low power requirement which was only about 10% of the current state-of-art massively parallel designs.

In the second session five talks were given on varied topics. Talk one, presented by Carstin Scholtes considered a direct mapped cache design and it then showed how source codes for sparse dense matrix multiply and sparse Cholesky factorization could be examined to derive a set of calculations that could be evaluated in constant time to yield a probabilistic determination of the number of cache misses to be expected when the program was executed on a specific architecture, [8]. The second talk, presented by Fred Gustavson, described a new algorithm for the level 3 BLAS $\text{DGEMM}$ on a computer with $M + 1$ levels of memory, $M$ caches and main memory. Its main contribution was to extend the basic two level algorithm $M = 1$ in two ways: (1) streaming was added to the basic two level algorithm and (2) the basic two level algorithm was shown to work at level $i + 1$ to level $i$ for $i = 1, \ldots, M$. These two ideas were combined with another idea: conservation of matrix data. The combination of these three ideas allowed one to show that one could reduce the number of basic $\text{DGEMM}$ algorithms from $6^{M+1}$ to just 4, [9]. Paper three, presented by Keshav Pingali was about a general purpose compiler that obtains high performance for some common dense linear algebra codes, [10]. This method was a deterministic method for finding key hardware parameters of an architecture as opposed to the AEOS approach of the ATLAS project, [11]. Paper four was on software testing of Library Software and in particular, the Lapack Library, [12]. It was presented by Tim Hopkins. This paper defines a quantitative measure of software quality (metrics) and a way to reduce the amount of software testing by allowing tests that fail to improve the metrics to be discarded. The paper observes how Lapack’s testing codes actually “measure up” in a software engineering sense. Their initial results indicate that many of Lapack’s test codes fail to improve the coverage of their metrics and that many sections of Lapack’s code are never executed by the testing software. Some of this lack of coverage is easily remedied; other parts require expertise about the codes being tested. The last paper by Peter Drackenberg is somewhat related to the theme of new data structures of DLA. His approach is to describe a data type for dense matrices whose primitive operations are decomposition and composition (of submatrices) as opposed to indexed element access which is the primitive operation on conventional arrays. This work appears to be related to the Lapack approach and recent work on recursion and iteration done by many authors. Performance results on SGI Octane Systems for level 3 BLAS were found to be encouraging.

Not all of these talks appears in these Lecture Notes. Only talk four of Session one, [5] and talks two, four and five of Session two, [9] appear. However, for completeness the following references [12] covers talks one to three of Session one. An excellent set of slides is available for talk five of Session one and talk one of Session two, [7,8]. [10] covers talk three of Session 2.