Auto-optimization of linear algebra parallel routines: the Cholesky factorization

Luis-Pedro García [*], Javier Cuenca [†], Domingo Giménez [‡]

Preferred Topic Area: Software Engineering

Abstract

In last years different techniques to obtain software able to tune automatically to the conditions of the execution platform have been studied [1, 2, 3, 4]. Such development will facilitate efficient utilization of the routines by non-expert users, e.g. those normally using linear algebra routines in the solution of large scientific or engineering problems.

In this paper we discuss the approach in [3] for modelling the behavior of linear algebra algorithms, with the aim of obtaining self-optimizing routines, and the technique is applied to the parallel routine for the Cholesky factorization. The factorization is obtained with a block-cyclic partitioning in a logical two-dimensional mesh of $p = r \times c$ processes (in ScaLAPACK style). An analytical model of the execution time of the parallel algorithm is developed as a function of the problem size, the system parameters (parameters of a target platform) and the algorithmic parameters. The algorithm is studied both theoretically and experimentally in order to determine the effect of the value of the system parameters on the algorithmic parameters. The typical system parameters considered in the study are the cost of arithmetic operations using BLAS kernels of levels 1, 2 or 3 (k_1, k_2, k_3) and the cost of the communication parameters (start-up, t_s, and word sending time, t_w) for the used MPI library. The algorithmic parameters are the block size b (a block based algorithm is considered) and the parameters r and c defining the logical topology of the process grid and data distribution.

In this paper we will see that in order to obtain a good estimation it is necessary to use different costs for different routines of the same level and to use different costs for different types of MPI communication mechanisms

[*]Servicio de Apoyo a la Investigación Tecnológica, Universidad Politécnica de Cartagena, 30203 Cartagena, SPAIN, luis.garcia@sait.upct.es

[†]Departamento de Ingeniería y Tecnología de Computadores, Universidad de Murcia, 30071 Murcia, SPAIN, javiercm@ditec.um.es

[‡]Departamento de Informática y Sistemas, Universidad de Murcia, 30071 Murcia, SPAIN, domingo@dif.um.es

Table 1: Parameters selection for the Cholesky factorization in different platforms.

	P4net $p=4$				HPC160smp $p=4$				HPC160mc $p=8$			
	opt.		mod.		opt.		mod.		opt.		mod.	
n	b	$r \times c$	b	$r \times c$	b	$r \times c$	b	$r \times c$	b	$r \times c$	b	$r \times c$
512	64	1×1	64	1×1	32	4×1	32	4×1	32	2×4	32	2×4
1024	128	1×1	128	1×1	64	4×1	64	4×1	32	2×4	32	2×4
2048	128	1×1	128	1×1	64	4×1	64	4×1	64	2×4	32	2×4
4096	256	1×1	256	1×1	128	4×1	128	4×1	128	2×4	128	2×4
5120	256	2×1	256	2×1	128	4×1	128	4×1	64	2×4	64	2×4

(user-defined datatypes or predefined datatypes). It will be necessary to take into account that the cost of the communication parameters varies according to the volume of the communication.

The execution time predicted by the model and the experimental execution time are compared for different matrix and block sizes, number of processors and logical mesh topologies. Experiments are carried out in a network of Pentium 4 with Fast-Ethernet (P4net) and in a network of HP AlphaServer quad processors using Shared Memory (HPC160smp) and MemoryChannel (HPC160mc). Table 1 shows the values of the algorithmic parameters provided by our method when following the model (mod.) and the experimental optimum values (opt.). The values of the parameters vary for different systems and problem sizes, but with the model a satisfactory selection of the parameters is made in all the cases. Only in HPC160mc for matrix size 2048 are the block sizes different. In this case the execution time obtained with the values provided by the model is about 14% higher than the optimum experimental time.

Acknowledgements. Partially supported by Spanish MCYT and FEDER under Grant TIC2003-08238-C02-02.

References

[1] Frigo, M.: FFTW: An Adaptive Software Architecture for the FFT. In Proceedings of the ICASSP Conference, volume 3, p 1381. 1998.

[2] Chen, Z., Dongarra, J., Luszczek, P., Roche, K.: Self Adapting Software for Numerical Linear Algebra and LAPACK for Clusters. Parallel Computing, 29 (11-12), Elsevier Science, pp. 1723-1743, 2003.

[3] J. Cuenca, D. Giménez and J. González, Architecture of an automatically tuned linear algebra library. Parallel Computing, 30 (2), Elsevier Science, pp. 187-210, 2004.

[4] Katagiri, T., Kise, K., Honda, H.: Effect of Auto-tuning with User's Knowledge for Numerical Software. Proceedings of ACM Computing Frontiers, 2004.