Efficient Calculation Managing on a Cluster with Distributed Memory

Evgeny Ilchenko and Gennadi Malaschonok

Abstract. Managing of cluster parallel computations for tree-like recursive algebraic algorithms for the case of cluster with distributed memory is one of the difficult problems of computer algebra. The block-recursive algorithms of matrix and polynomial multiplication, Strassen's and Karatsuba's algorithms, matrix inversion and computation of the kernel of a matrix operator, LDU and Bruhat factorization are examples of such algorithms. We suggest a scheme with multidispatching for management of such parallel computing processes and demonstrate the results of experiments at the JSC RAS cluster MVS-10P.

Introduction

The task of managing calculations on a cluster with distributed memory for sparse matrix algorithms is today one of the most difficult challenges [1]. This task can not be solved independently of the algorithm itself, since it can not be solved in an abstract setting. With any approach to such a task, it is necessary to somehow fix the class of algorithms.

We consider the class of block-recursive matrix algorithms. The most famous of them are standard and Strassen's block matrix multiplication, Schur and Strassen's block-matrix inversion [2].

For such algorithms we suggest a scheme with multidispatching management of parallel computing processes. We demonstrate the results of experiments at the cluster computer MVS-10P. These experiments show high efficiency of this conception of management.

Class of block-recursive matrix algorithms

Generalization of Strassen's algorithm for triangular factorization and matrix inversion with permutations of rows and columns by J. Bunch and J. Hopkroft [3] is not a block-recursive algorithm. Block-recursive algorithms were not so important as long as the calculations were performed on computers with shared memory. Only in the nineties it became clear that block-recursive matrix algorithms are required to operate with sparse super large matrices on a supercomputer.

The block recursive algorithm for the solution of systems of linear equations and for adjoint matrix computation which is some generalisation of Schur inversion in commutative domains was discraibed in [4] and [5]. See also at the book [6]. However, in all algorithms, except matrix multiplication algorithms, there is a very strong restriction that is superimposed on the matrix. The leading minors of the matrix, which are on the main diagonal, should not be zero.

This restriction was removed later in the papers [7] - [9]. The algorithm that computes the adjoint matrix, the echelon form, and the kernel of the matrix operator for the commutative domains was proposed in [7]. The block-recursive algorithm for the Bruhat decomposition and the LDU decomposition for the matrix over the field was obtained in [8], and these algorithms generaized for the matrices over commutative domains was obtained in [9] and [10].

1. Other benefits and application

Control systems.

In 1967 Howard H. Rosenbrock introduced a useful state-space representation and transfer function matrix form for control systems, which is known as the Rosenbrock System Matrix [11]. Since that time, the properties of the matrix of polynomials being intensively studied in the literature of linear control systems.

Groebner basis.

Another important application is the calculation of Gröbner bases. A matrix composed of Buchberger S-polynomials is a strongly sparse matrix. Reduction of the polynomial system is performed when calculating the echelon and diagonal forms of this matrix. The algorithm F4 was the first such matrix algorithm.

Solving PDE's for particle interaction.

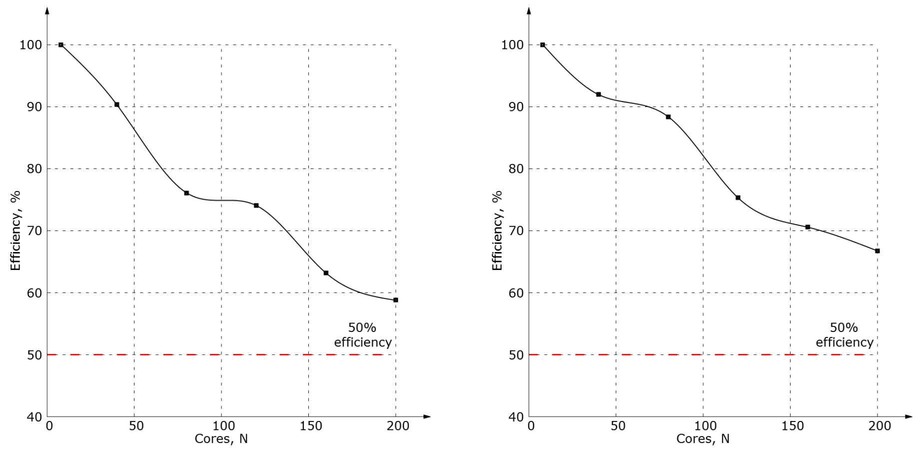
The conservation of the matrix sparseness during the Bruhat decomposition was first investigated in [12]. One of the important class of sparse matrix is called quasiseparable. Any submatrix of quasiseparable matrix entirely below or above the main diagonal has small rank. These quasiseparable matrices arise naturally in solving PDE's for particle interaction with the Fast Multi-pole Method (FMM). The efficiency of application of the block-recursive algorithm of the Bruhat decomposition to the quasiseparable matrices is studied in the article [13].

2. Calculation managing and experiments

The block-recursive matrix algorithms for sparse matrix require a special approachs to managing parallel programs. One approach to the cluster computations management is a scheme with one dispatcher (or one master).

We consider another scheme of cluster menagement. It is a scheme with multidispatching, when each involved computing module has its own dispatch thread and several processing threads[15, 14].

Let us denote by N_i the number of cluster cores and by T_i the computational time in the i-th experiment. For the theoretical best case we'd like to have constant product: $\forall i : \ T_i N_i = \mathbf{const}$. So to demonstrate the efficiency of parallel computational process we have to know the value $E_i = 100\% \ T_i N_i/(T_1 N_1)$. This is the efficiency which was demonstrated in the i-th experiment.



matrix size is 8192 matrix size is 16384

FIGURE 1. The efficiency of the calculation of Schur matrix inversion as a function of the number of cores.

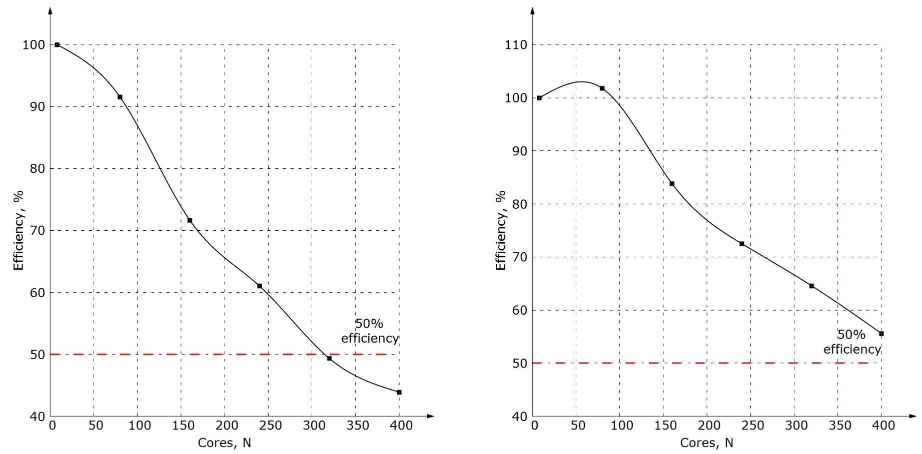
Let $A = \begin{pmatrix} A_0 & A_1 \\ A_2 & A_3 \end{pmatrix}$, $\det(A_0) \neq 0$, $\det(A) \neq 0$, then Schur-Strassen block-recursive matrix inversion is described by formula:

$$A^{-1} = \begin{pmatrix} \mathbf{I} & -A_0^{-1}A_1 \\ 0 & \mathbf{I} \end{pmatrix} \times \begin{pmatrix} \mathbf{I} & 0 \\ 0 & (A_3 - A_2A_0^{-1}A_1)^{-1} \end{pmatrix} \times \begin{pmatrix} \mathbf{I} & 0 \\ -A_2 & \mathbf{I} \end{pmatrix} \times \begin{pmatrix} A_0^{-1} & 0 \\ 0 & \mathbf{I} \end{pmatrix}.$$

For standard matrix multiplication we call it Schur method. And we call it Schur-Strassen method when we used here Strassen's matrix multiplication.

The results of experiments with Schur matrix inversion algorithm are shown in fig. 1. The results for computation of kernel of the matrix operator and adjoint matrix [7] are shown in fig. 2. We use matrices over finite numerical fields in both cases.

The computational experiments was done in Joint Supercomputer Center of the Russian Academy of Sciences (http://www.jscc.ru/scomputers.html) at the cluster mvs10p: Intel Xeon E5-2690, 64G RAM Intel(R) MPI Library for Linux* OS, Version 4.1.0.



matrix size is 8192 matrix size is 12288

FIGURE 2. The efficiency of the calculation of kernel and adjoint matrix as a function of the number of cores.

In experiments we made calculations for identical matrices, using a different number of cluster's nodes.

For the Schur inversin algorithm the minimum number of cores was 8 (1 node), and the maximum number of cores was 200 (25 nodes). For the matrix of size 8192x8192 the number of cores increased 25 times and as a result the calculation time changed 15 times: from 749 sec. to 51 sec. For the matrix of size 16384x16384 number of cores increased 25 times and the calculation time changed 17 times: from 1159 sec. to 69 sec.

For our adjoint and kernell computation algorithm the minimum number of cores was 8 (1 node), and the maximum number of cores was 400 (50 nodes). For the matrix of size 8192x8192 the number of cores in experiment increased 50 times and as a result the calculation time changed 22 times: from 3969 sec. to 181 sec. For the matrix of size 12288x12288 the number of cores increased 50 times and the calculation time changed 28 times: from 15249 sec. to 548 sec.

Static Control of a Parallel Computing Process

One of the most popular approaches in computer algebra is the method of homomorphic images, based on the Chinese remainder theorem. It allows solving the problem simultaneously in many factor rings and using these solutions to obtain a solution of the original problem. For such a process of parallelization, it is sufficient to use static control of calculations on a cluster.

If the elements $m_i, i = 1, .., k$, of the commutative ring R are relatively prime, $m_1..m_k = \mu$, $r_i = x \bmod m_i, i = 1, .., k$, then to find the $x \bmod \mu$ the well-known Newton scheme is usually used.

Below we give Newton scheme for the particular case where the main ring R is a ring of integer numbers. In contrast to the standard scheme of calculations, we only change the order of actions, but we get fewer operations.

Let n_{ij} be inverse element of m_i in Z/m_jZ, i.e. $n_{ij}m_i = 1 \bmod m_j$. Denote:

$$c_1 = r_1, \quad \bar{c}_1 = c_1 \bmod m_2, \; \nu_1 = n_{1,2} \bmod m_2,$$
$$c_2 = c_1 + m_1(\nu_1(r_2 - \bar{c}_1) \bmod m_2), \; \bar{c}_2 = c_2 \bmod m_3, \; \nu_2 = n_{1,3}n_{2,3} \bmod m_3,$$
$$c_3 = c_2 + m_1m_2(\nu_2(r_3 - \bar{c}_2) \bmod m_3), \bar{c}_3 = c_3 \bmod m_4, \nu_3 = n_{1,4}n_{2,4}n_{3,4} \bmod m_4,$$
$$...$$
$$c_k = c_{k-1} + m_1...m_{k-1}(\nu_{k-1}(r_k - \bar{c}_{k-1}) \bmod m_k), \quad \bar{c}_k = c_k \bmod m_{k+1},$$
$$\nu_k = n_{1,k+1}n_{2,k+1}..n_{k,k+1} \bmod m_{k+1},$$
$$...$$
$$c_n = c_{n-1} + m_1...m_{n-1}(\nu_{n-1}(r_n - \bar{c}_{n-1}) \bmod m_n).$$

It is easy to verify by induction that the following inequalities hold

$$0 \leq c_k < m_1..m_k, \; k = 1, 2, ..n.$$

So the number c_n is not required to be modulo μ, since it is already in the required interval.

Of course, the same scheme can be successfully applied in the ring of polynomials and in all cases when the Chinese remainder theorem holds.

Example:

$$m_i = \{3_1, 5_2, 7_3\}, \; \mu = 105, x = 73,$$
$$r_i = \{1_1, 3_2, 3_3\}, n_{1,2} = 2, n_{1,3} = 5, n_{2,3} = 3, \nu_i = \{2_1, 1_2\},$$
$$c_1 = 1, \; \bar{c}_1 = 1,$$
$$c_2 = 1 + 3(2(3 - 1) \bmod 5) = 13, \; \bar{c}_2 = 6,$$
$$c_3 = 13 + 3 \cdot 5(1(3 - 6) \bmod 7) = 73.$$

References

[1] Dongarra J. With Extrim Scale Computing the Rules Have Changed. In Mathematical Software. ICMS 2016, 5th International Congress, Proceedings (G.-M. Greuel, T. Koch, P. Paule, A. Sommese, eds.), Springer, LNCS, volume 9725, pp. 3-8, 2016.

[2] Strassen V. Gaussian Elimination is not optimal. Numerische Mathematik. V. 13, Issue 4, 354–356 (1969)

[3] Bunch J., Hopkroft J. Triangular factorization and inversion by fast matrix multiplication. Mat. Comp. V. 28, 231-236 (1974)

[4] Malaschonok G. Recursive Method for the Solution of Systems of Linear Equations. Computational Mathematics. A. Sydow Ed, Proceedings of the 15th IMACS World Congress, Vol. I, Berlin, August 1997), Wissenschaft & Technik Verlag, Berlin 1997, 475-480.

[5] Malaschonok G. Effective Matrix Methods in Commutative Domains, Formal Power Series and Algebraic Combinatorics, Springer, Berlin, 2000, 506-517.

[6] Malaschonok G. Matrix computational methods in commutative rings. Tambov, TSU, 2002, 213 p.

[7] Malaschonok G. On computation of kernel of operator acting in a module // Vestnik Tambovskogo universiteta. Ser. Estestvennye i tekhnicheskie nauki [Tambov University Reports. Series: Natural and Technical Sciences], vol. 13, issue 1, 2008. 129-131.

[8] Malaschonok G. Fast Generalized Bruhat Decomposition. Computer Algebra in Scientific Computing, LNCS 6244, Springer, Berlin 2010. 194-202. DOI 10.1007/978-3-642-15274-0_16. arxiv:1702.07242

[9] Malaschonok G. Generalized Bruhat decomposition in commutative domains /in book: Computer Algebra in Scientific Computing. CASC'2013. LNCS 8136, Springer, Heidelberg, 2013, 231-242. DOI 10.1007/978-3-319-02297-0_20. arxiv:1702.07248

[10] Malaschonok G., Scherbinin A. Triangular Decomposition of Matrices in a Domain. Computer Algebra in Scientific Computing. LNCS 9301, Springer, Switzerland, 2015, 290-304. DOI 10.1007/978-3-319-24021-3_22. arxiv:1702.07243

[11] Rosenbrock, H.H. Transformation of linear constant system equations. Proc. I.E.E. V.114, 541–544. (1967).

[12] Kolotilina L., and A. Yemin A. Bruhat decomposition and solution of linear algebraic systems with sparse matrices. Sov.J.Numer.Anal. and Math.Model. v.2, 421–436 (1987).

[13] Pernet C., Storjohann A. Time and space efficient generators for quasiseparable matrices. arXiv:1701.00396 (2 Jan 2017) 29 p.

[14] Ilchenko E.A. An algorithm for the decentralized control of parallel computing process // Vestnik Tambovskogo universiteta. Ser. Estestvennye i tekhnicheskie nauki [Tambov University Reports. Series: Natural and Technical Sciences], 2013, vol. 18, issue 4, 1198-1206.

[15] Ilchenko E.A. About effective methods parallelizing block recursive algorithms // Vestnik Tambovskogo universiteta. Ser. Estestvennye i tekhnicheskie nauki [Tambov University Reports. Series: Natural and Technical Sciences], 2015, vol. 20, issue 5, 1173-1186.

Evgeny Ilchenko
Tambov State University named after G.R.Derzhavin
Tambov, Russia
e-mail: `ilchenkoea@gmail.com`

Gennadi Malaschonok
Tambov State University named after G.R.Derzhavin
Tambov, Russia
e-mail: `malaschonok@gmail.com`