Parallel Computation of Matrix Discrete Multiplications in the Field of Differential Transformations Based on Work-Stealing Task Scheduler*

Hamlet, Aslanyan
National Polytechnic University of Armenia
Yerevan, Armenia
e-mail: aslanian.hamlet@gmail.com

Sargis, Simonyan
National Polytechnic University of Armenia
Yerevan, Armenia
e-mail: ssimonyan@seua.am

ABSTRACT
A method for parallel computation of matrix discrete multiplications in the field of differential transformations is presented based on work-stealing task scheduler. The application of the approach on a method of determining Drazin parametric generalized inverse matrices is given. A comparative analysis of serial and parallel versions of the methods applied on randomly generated test matrices is presented.

Keywords
Information technology, differential transformations, parallel computation, work-stealing task scheduler, matrix discretes, Drazin parametric generalized inverse matrices.

1. INTRODUCTION
Differential transformations [1] are defined as follows:

$$X(K) = \frac{H^K}{K!} \cdot \frac{\partial^K x(t)}{\partial t^K}\Big|_{t=t_v}, K = \overline{0,\infty} \quad \underline{\overline{\bullet}} \quad x(t) = \aleph(t, t_v, H, X(K))$$

where t can be time, Laplace operator ($S \sim \frac{d}{dt}$) or any other parameter, K is an integral argument, H - scale factor, t_υ - the center of approximation, $X(K)$ - the K-th discrete of the original $x(t)$, and $\underline{\overline{\bullet}}$ is the sign of transfer from the field of originals to the field of differential imprints, and vice versa.

These transformations found their application in a wide range of spheres like automated and optimal control [2, 3], differential equations [4, 5], mathematical programming [2], analysis of parametric matrices [6], determining of parametric generalized inverse matrices [7, 8, 9], etc.

In the field of differential transformations, matrix discretes of the parametric matrix $A(t)$ are defined as follows:

$$A(K) = \frac{H^K}{K!} \cdot \frac{\partial^K A(t)}{\partial t^K}\Big|_{t=t_v}, K = \overline{0,\infty} \quad \underline{\overline{\bullet}} \quad A(t) = \aleph(t, t_v, H, A(K)).$$

The multiplication of matrix discretes $A(K), K = \overline{0,\infty}$ and $B(K), K = \overline{0,\infty}$ looks like [1, 2]:

$$C(k) = A(k) * B(k) = \sum_{l=0}^{k} A(l)B(K-l), k = \overline{0,K}.$$

Or in more detail:

$$C(0) = A(0) \cdot B(0),$$
$$C(1) = A(1) \cdot B(0) + A(0) \cdot B(1),$$
$$\dots$$
$$C(K) = A(K) \cdot B(0) + A(K-1) \cdot B(1) + \cdots + A(1) \cdot B(K-1) + A(0) \cdot B(K).$$

Despite recent advances in the development of fast algorithms for matrix multiplication [10, 11, 12], it remains a heavy computational spot. In this paper we assume that the complexity of matrix multiplication is $O(mnl)$, where mxn and nxl are the sizes of multiplicands. The complexity of the matrix discrete multiplication is $\frac{(K+1)(K+2)}{2}$ times more than $O(mnl)$. These multiplications occur very frequently during various calculations in the field of differential transformations and, hence, speeding up their computational time by performing the calculations in parallel will have a positive impact on all the methods which make use of them.

Matrix arithmetic operations, particularly multiplication, are available through libraries and APIs like BLAS [13] (including a portable implementation of BLAS interface – ATLAS [14]), LAPACK [15], Intel MKL [16], etc.

In this paper an algorithm for parallel realization of matrix discrete multiplications is presented, based on the concept of work-stealing task scheduler [16, 17] which is an effective way to exploit parallelism [18] and is advanced by industry leaders such as Intel [18, 19]. The basic concept of the work-stealing task scheduler is as follows [17]:

1. A queue of tasks is maintained for each thread of execution. New tasks are added at the end of the queue.
2. Each thread pulls a task to execute from the back of its task-queue (where the tasks are probably still "hot" in cache).
3. If the thread drains its task-queue, it steals a task from the top of another thread's task-queue.

As a result, work-stealing task scheduler ensures a constant workload for all the threads of execution. It is to be noted though that this scheduler proves to be efficient when all the tasks have approximately the same computational difficulty [16].

Intel Threading Building Blocks library [18] is considered as an instance of a work-stealing task scheduler in this paper. This library allows having an exploitation of parallelism programmatically, during the execution of the program, using a provided API and linking with its implementation library, whereas another widespread approach of parallel realization – OpenMP, is a language extension (particularly C and Fortran) and consists of compiler directives targeted to generate a parallel-execution-oriented program [16, 20]. A similar extension to C and C++ languages is "Cilk Plus" [16, 20]. In comparison, Intel Threading Building Blocks is targeted to be used in programs written in C++, is free from type restrictions present in OpenMP, and provides a larger set of parallel algorithms and means of realization of parallel design patterns than OpenMP [16, 18, 20].

2. DESCRIPTION OF THE METHOD

Based on the definition, the process of matrix discrete multiplications in the field of differential transformations is composed of separate and independent operations for each $k \in \overline{0,K}$. Hence, it can be parallelized. The method of parallel realization of matrix discrete multiplications based on the number of primitive operations per thread is as follows:

Step 1: The number of primitive operations (multiplications) n_0 for each task is chosen. Based on n_0, the number of matrix multiplications by each task is determined as $p = \frac{n_0}{mnl}$.

Step 2: Set $k = 0$.

Step 3: A task is created for the range (k, x), where x is such that the number of matrix multiplications in the range (k, x) is p. It is easy to prove that

$$x = \frac{-3 + \sqrt{9 - 4(2 - (k^2 + 3k + 2 + 2p))}}{2}.$$

Step 4: $k = x + 1$.

Step 5: If $k \le K$, switch to the step 3. Otherwise, end the algorithm.

In this case each thread will execute approximately the same n_0 number of t_0 operations. As a result, on an ideal machine (where all timing delays are discarded) with c number of cores, the time spent on matrix discrete parallel multiplication is defined as

$$T_{parallel} = \frac{(K+1)(K+2)mnl}{2n_0 c} \cdot n_0 t_0 = \frac{(K+1)(K+2)mnl}{2c} \cdot t_0.$$

Whereas for the serial version we have

$$T_{serial} = \frac{(K+1)(K+2)mnl}{2} \cdot t_0.$$

The number of multiplications n_0 per each thread is chosen based on the matrix dimensions, K and the number of cores available on the machine.

3. EXPERIMENTS

We apply the proposed approach on parallel realization of matrix discrete multiplications on the example of a method for determining Drazin parametric generalized inverse matrices [21, 22]. Drazin parametric generalized inverse matrix $A^D(t)$ of a square matrix $A(t)$ of size n is defined as:

$$A^D(t)A(t)A^D(t) = A^D(t),$$
$$A(t)A^D(t) = A^D(t)A(t),$$
$$A^{k+1}(t)A^D(t) = A^k(t),$$

where $k = index(A(t))$ is the lowest non-negative number for which $rank(A^{k+1}(t)) = rank(A^k(t))$ and is called the index of a matrix [22]. The method for determining Drazin parametric generalized inverse matrices in the field of differential transformations looks as follows [7]:

Step 1: The following matrix discretes are calculated:

$$A(K) = \frac{H^K}{K!} \cdot \frac{\partial^K A(t)}{\partial t^K}\Big|_{t=t_v}, K = \overline{0,\infty},$$

$$A^n(K) = A^{n-1}(K) * A(K) = \sum_{l=0}^{K} A^{n-1}(l)A(K-l), K = \overline{0,\infty}.$$

Step 2: The matrix discretes of the skeleton decomposition $A^n(t) = B(t) \cdot C(t)$ are calculated [23], where $B(t)$ and $C(t)$ are of sizes nxr and rxn, respectively, and r is the rank of the matrix $A(t)$.

Step 3: By setting $X(t) = C(t)B(t)$, the matrix discretes $X^{(-1)}(K), K = \overline{0,\infty}$ of $X^{-1}(t)$ are calculated [24] (note that $X^{(-1)}(K)$, is the K–th matrix discrete of the inverse matrix $X^{-1}(t)$, and not the inverse matrix of the K–th matrix discrete $X(K)$).

Step 4: The matrix discretes of Drazin parametric generalized inverse matrix $A^D(t)$ are calculated:

$$A^D(K) = A^{n-1}(K) * B(K) * X^{(-2)}(K) * C(K).$$

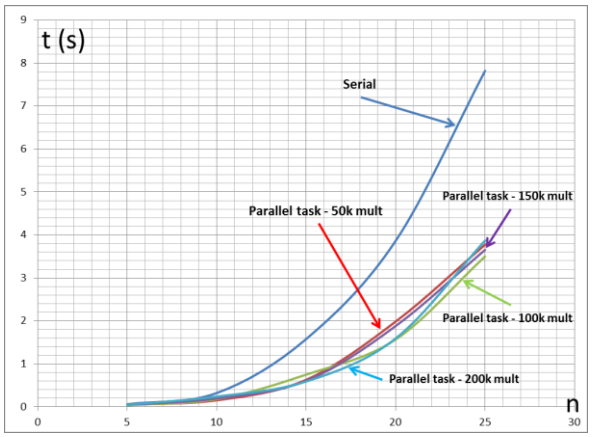
As it is the case with most of the methods in the field of differential transformations, the original $A^D(t)$ can then be restored from the matrix discretes $A^D(K)$ by means of several well-known backward restoration transformations like single- and multi- point differential-Taylor, Pade transformations, etc. [1, 2]

The serial and parallel implementations of the method in C++ were tested against the following test conditions:
1. Intel Threading Building Blocks was taken as an example of work-stealing task scheduler.
2. Randomly generated polynomial test square matrices of sizes ranging from 5 to 25 were fed to the input of the method. The highest degree of polynomials was 5.
3. The number of multiplications n_0 per each thread was chosen as 50 000, 100 000, 150 000 and 200 000.
4. Test machine was 4-core Intel(R) Core(TM) i5-4210U CPU @ 1.70GHz.
5. The number of discretes was taken as $K = 100$.

The result of serial and parallel executions of the method is presented in the table below (in seconds), where the columns denote the size of the input matrix, the time of serial execution and the times of parallel executions of the method for each number of multiplications n_0 per each thread.

Size	Serial	Parallel 50 000	Parallel 100 000	Parallel 150 000	Parallel 200 000
5	0.063	0.049	0.041	0.046	0.048
10	0.323	0.154	0.180	0.197	0.240
15	1.579	0.631	0.751	0.607	0.592
20	3.863	1.985	1.569	1.881	1.592
25	7.819	3.776	3.502	3.654	3.871

The graphical representations of the same results are presented in the figure below. All measurements of time are in seconds.



The choice of the number n_0 is usually not obvious and is done by performing several experiments. Based on the results of the experiments presented here, the choice of $n_0 = 100000$ as a number of primitive multiplications per each thread, in average, provides the best results for this method of determining Drazin parametric generalized inverse matrices – more than 2 times performance gain for parallel realization.

4. CONCLUSION

A method for parallel realization of matrix discrete multiplications in the field of differential transformations is presented based on the concept of work-stealing task scheduler. Taking into account the fact that the computational difficulty of each parallel task in this approach is relatively the same, this method proves to be very efficient in conjunction with this type of scheduler which is proven by the theoretical analysis of it on an ideal machine. The approach is applied in the process of determining Drazin parametric generalized inverse matrices. Test results of the realization of the method in C++ on randomly generated test matrices explicitly showed the advantages of the parallel realization and provided data for the choice of the number n_0 of primitive operations per thread.

REFERENCES

[1] Г. Пухов, "Дифференциальные преобразования функций и уравнений", *Наукова думка*, 419с., 1980.

[2] С. Симонян, А. Аветисян, "Прикладная теория дифференциальных преобразований", *Изд-во ГИУА "Чартарагет"*, 361с., 2010.

[3] I. Hwang, J. Li, D. Du, "A Numerical Algorithm for Optimal Controls of Class of Hybryd Systems: Differential Transformation Based Approach", *International Journal of Control, Vol. 81, N2*, pp. 277-293, 2008.

[4] K. Batiha, B. Batiha, "A New Algorithm for Solving Linear Ordinary Differential Equations", *World Applied Sciences Journal 15, N12*, pp. 1774-1779, 2011.

[5] M. Eslami, H. Zareamoghaddam, "Differential Transform Method for Abel Differential Equation", *World Applied Sciences 13, N5*, pp. 1005-1011, 2011.

[6] М. Тамазян, "Разработка методов определения характеристик неавтономных матриц и автоматизация вычислительных процедур", *Автореф. дисс. ... к.т.н.*, 24с., 2012.

[7] Г. Асланян, С. Симонян, "Д-аналог метода определения однопараметрической обобщенной обратной матрицы Дразина, основанный на скелетном разложении матрицы", *Известия Томского политехнического университета, T.325, N2*, С.29-34, 2014.

[8] С. Симонян, "Определение квадратных параметрических обобщенных обратных матриц Мура-Пенроуза применением дифференциальных преобразований Пухова", *Известия Томского политехнического университета, T.323, N2*, С.6-10, 2013.

[9] С. Симонян, "Параллельные вычислительные методы определения параметрических обобщенных обратных матриц", *Известия Томского политехнического университета, Т. 323, N5*, С. 10-15, 2013.

[10] V. Strassen, "Gaussian elimination is not optimal", *Numerische Mathematik 14, N3*, pp. 354–356, 1969.

[11] D. Coopersmith, S. Winograd, "Matrix Multiplicaton via Arithmetic Progressions", *J. Symbolic Computation*, pp. 251-280, 1990.

[12] F. Le Gall, "Powers of Tensors and Fast Matrix Multiplication", *Proceedings of the 39th International Symposium on Symbolic and Algebraic Computation*, pp. 296-303, 2014.

[13] http://www.netlib.org/blas/

[14] http://math-atlas.sourceforge.net/

[15] E. Anderson, Z. Bai, C. Bischof, S. Blackford, et al, "LAPACK Users' Guide (Software, Environments and Tools)", 3rd edition, *SIAM*, 429p., 1987.

[16] M. McCool, J. Reinders, A. Robison, "Structured Parallel Programming: Patterns for Efficient Computation", *Morgan Kaufmann*, 432 p., 2012.

[17] R. Blumofe, C. Leiserson, "Scheduling Multithreaded Computations by Work-Stealing", *Proceedings of the 35th Annual IEEE Conference on Foundations of Computer Science*, 29p., 1994.

[18] J. Reinders, "Intel Threading Building Blocks: Outfitting C++ for Multi-core Processor Parallelism", *O'REILLY*, 336p., 2007.

[19] A. Kukanov, M. Voss, "The Foundations for Scalable Multi-core Software in Intel Threading Building Blocks", *Intel Technology Journal, Vol. 11, Issue 4*, pp. 309-322, 2007.

[20] J. Jeffers, J. Reinders, "Intel Xeon Phi Coprocessor High-Performance Programming", *Morgan Kaufmann*, 432p., 2013.

[21] M. Drazin, "Pseudo-inverses in associative rings and semigroups", *The American Mathematical Monthly 65(7)*, pp. 506–514, 1958.

[22] S. Campbell, C. Meyer, "Generalized Inverses of Linear Transformations", *Society for Industrial and Applied Mathematics (SIAM)*, 292 p., 2008.

[23] С. Симонян, Г. Асланян, "Метод определения параметрических (B,Q)- обобщенно-обратных матриц", *Известия НАН РА и ГИУА, Сер. ТН, Т. LXVII, N2*, С.220-226, 2014.

[24] С. Симонян, М. Тамазян, "Д-аналог L(t)U(t)-разложения для обращения неавтономных матриц", *Вестник ГИУА. Серия "Информационные технологии, электроника, радиотехника", Вып. 15, N1*, С.35-41, 2012.