Parallel computations based on automatic transformation of ordinary differential equations

Jan Kopiva, Jií Kunovský, Václav Šátek, Martina Drozdová, and Alexander Schirrer

Citation: AIP Conference Proceedings **1558**, 2293 (2013); doi: 10.1063/1.4825998
View online: http://dx.doi.org/10.1063/1.4825998
View Table of Contents: http://scitation.aip.org/content/aip/proceeding/aipcp/1558?ver=pdfcov
Published by the AIP Publishing

Parallel Computations Based on Automatic Transformation of Ordinary Differential Equations

Jan Kopřiva*, Jiří Kunovský*, Václav Šátek†, Martina Drozdová* and Alexander Schirrer**

*University of Technology, Faculty of Information Technology, Božetěchova 2, 612 66 Brno, Czech Republic [1]

†IT4Innovations, VŠB Technical University of Ostrava, 17. listopadu 15/2172, 708 33 Ostrava-Poruba, Czech Republic [2]

**Vienna University of Technology, Karlsplatz 13, 1040 Vienna, Austria

Abstract. The paper is a part of student cooperation in AKTION project (Austria-Czech). Theoretical work on the numerical solution of ordinary differential equations by the Taylor series method has been going on for a number of years. The simulation language TKSL has been created to test the properties of the technical initial problems and to test an algorithm for Taylor series method [1].

The Residue Number System (RNS) has great potential for accelerating arithmetic operation, achieved by breaking operands into several residues and computing with them independently.

Keywords: Ordinary Differential Equations, Taylor Series Method, TKSL, Residue Number System, Montgomery Multiplication
PACS: 02,89

INTRODUCTION

Methods of numerical solutions of ordinary differential equations (ODEs) have been studied since the end of the last century. A large number of integration formulas have been published especially for solving special systems of differential equations. In general, it was not possible to choose the best method but for a subclass of tasks defined by similar properties the most suitable method could always be found. "Modern Taylor Series Method" has proved to be both very accurate and fast. It is based on a direct use of the Taylor series.

The Modern Taylor Series Method used in the computations increases the method order ORD automatically, i.e. the values of the Taylor series terms are computed for increasing order until adding the next term does not improve the accuracy of the solution.

The main problem connected with using Taylor series is the need to generate higher derivatives. This is in fact the reason why Runge-Kutta formulas of various orders have been used. If we succeed, however, to obtain the terms with higher derivatives, the accuracy of calculations by Taylor series method is extreme (it is in fact only limited by the type of the arithmetic unit used). This is typical, in particular, of the solution of the technical initial problems.

Technical initial problems are defined as initial problems where the right-hand side functions of the system are those occurring in the technical practice, that is functions generated by adding, multiplying and superposing elementary functions. Such systems can be expanded into systems with polynomials on the right-hand sides of the equations. In such a case the Taylor series terms can easily be calculated.

The Residue Number System (RNS) has great potential for accelerating arithmetic operation, achieved by breaking operands into several residues and computing with them independently.

Many of the fundamental difficulties with RNS arithmetic have been overcome (like fast conversion from RNS to binary [2],[3] or comparsion [4]) and RNS could offer a good accelaration of computation. Most implementations are based on FPGA and are used for error detection [5], FIR filters [6], cryptography [7] and another fields of electronics.

[1] kunovsky@fit.vutbr.cz
[2] vaclav.satek@vsb.cz

AUTOMATIC TRANSFORMATION

In this article we are concerned with initial problems described by autonomous systems of differential equations. We can often observe that, if the functions on the right-hand sides of the autonomous systems of differential equations are of a particular type frequently encountered in engineering applications, a sequence of substitutions can be found that transforms the original system into a new system with polynomials on the right-hand sides [8].

As an example a simple transformation of an initial problem

$$y' = y + \sin(t) \qquad y(0) = y_0 \tag{1}$$

is presented.

New initial problem after the transformation can be written in the form

$$\begin{aligned} y' &= y + y_1 \qquad y(0) = y_0 & (2)\\ y_1' &= y_2 & (3)\\ y_1' &= y_2 \qquad y_1(0) = 0 & (4)\\ y_2' &= -y_1 \qquad y_2(0) = 1 & (5) \end{aligned}$$

Coresponding block diagram is in Fig. 1. Fig. 1 represents a parallel cooperation of numerical integrators.

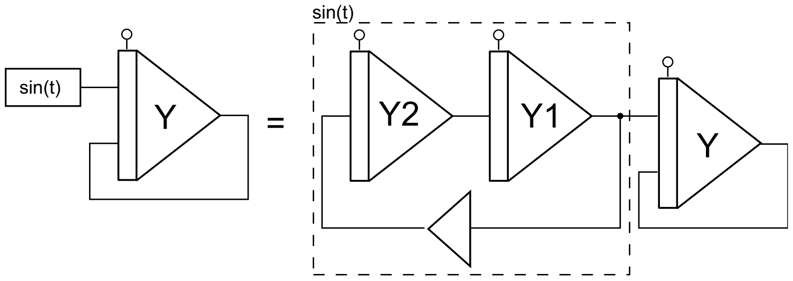

FIGURE 1. Corresponding block diagram

Basic demonstration of the elementary function transformation of $y_1 = \sin(t)$ follows.

$$\begin{aligned} y_1' &= \cos(t) \qquad y_1(t_0) = \sin(t_0)\\ y_2 &= \cos(t)\\ y_2' &= -\sin(t) \quad y_2(t_0) = \cos(t_0)\\ y_2' &= -y_1 \end{aligned}$$

Higher derivatives are

$$\begin{aligned} y_1'' &= (y_1')' = (y_2)' = y_2' = -y_1\\ y_1''' &= (y_1'')' = -y_1' = -y_2 \end{aligned}$$

Elementary function transformation is based on the identity of the Taylor series of functions $\sin(t)$ and $y_1(t)$:

$$\begin{aligned} \sin(t+h) &= \sin(t) + h\sin'(t) + \tfrac{h^2}{2!}\sin''(t) + \tfrac{h^3}{3!}\sin'''(t) + \cdots =\\ &= \sin(t) + h\cos(t) + \tfrac{h^2}{2!}(-\sin(t)) + \tfrac{h^3}{3!}(-\cos(t)) + \cdots \end{aligned}$$

$$\begin{aligned} y_1(t+h) &= y_1(t) + hy_1'(t) + \tfrac{h^2}{2!}y_1''(t) + \tfrac{h^3}{3!}y_1'''(t) + \cdots =\\ &= y_1(t) + hy_2(t) + \tfrac{h^2}{2!}(-y_1(t)) + \tfrac{h^3}{3!}(-y_2(t)) + \cdots =\\ &= \sin(t) + h\cos(t) + \tfrac{h^2}{2!}(-\sin(t)) + \tfrac{h^3}{3!}(-\cos(t)) + \cdots \end{aligned}$$

Of course, Modern Taylor Series Method must be used for computation of the new (transformed) intial problem - using as many as possible Taylor series terms.

THE IDEA OF PARALLEL COMPUTATIONS

This paper concentrates on large systems of numerical integrators. The idea of this approach comes from analogue methods of computations. Analogue methods are basically parallel methods and their analysing shows that independent parallel cooperation of multiple processors may be implemented by applying differential calculus. Using analogue methods, independent parallel cooperation of microprocessors is going to be effective if each microprocessor is going to be numerically integrating.

The algorithms of parallel cooperation of numerical integrators can be derived from one step of numerical solutions. This parallel cooperation of independent numerical integrators may be completed using arbitrary chosen numerical integration formula including Euler's method, 2nd, 3rd and 4th order Runge-Kutta method, Taylor series method etc.

INDEPENDENT COMPUTATIONS IN NUMERICAL INTEGRATORS

As an example, parallel cooperation of solution of

$$y'' + y = \quad 0 \; y(0) = 1,\; y'(0) = 0 \tag{6}$$

is presented.

The well known transformation of the 2^{nd} order differential equation into a set of two 1^{st} order differential equations is in (4),(5)

The equations for independent calculation of the first step of differential equation (6) using Euler (E), Runge-Kutta (R-K of the 2^{th} order, 4^{th} order) integration formulas for independent integrators Y1, Y2 (Fig. 1) are shown in Tab. 1.

TABLE 1. Formulas for solving 2^{nd} order differential equation (6)

E	$y_{1,0} = y_1(0)$	**E**	$y_{2,0} = y_2(0)$
	$K_0 y_{1,0} = h(y_{2,0})$		$K_0 y_{2,0} = h(-y_{2,0})$
	$y_{1,1} = y_{1,0} + K_0 y_{1,0}$		$y_{2,1} = y_{2,0} + K_0 y_{1,0}$
R-K	$y_{1,0} = y_1(0)$	**R-K**	$y_{2,0} = y_2(0)$
2^{nd} order	$K_0 y_{1,0} = h(y_{2,0})$	2^{nd} order	$K_0 y_{2,0} = h(-y_{1,0})$
	$K_1 y_{1,0} = h(y_{2,0} + K_0 y_{2,0})$		$K_1 y_{2,0} = h(-(y_{1,0} + K_0 y_{1,0}))$
	$y_{1,1} = y_{1,0} + \frac{K_0 y_{1,0}}{2} + \frac{K_1 y_{1,0}}{2}$		$y_{2,1} = y_{2,0} + \frac{K_0 y_{2,0}}{2} + \frac{K_1 y_{2,0}}{2}$
4^{th} order	$y_{1,0} = y_1(0)$	4^{th} order	$y_{2,0} = y_2(0)$
	$K_1 y_{1,0} = h(y_{2,0} + \frac{K_0 y_{2,0}}{2})$		$K_1 y_{2,0} = h(-(y_{1,0} + \frac{K_0 y_{1,0}}{2}))$
	$K_2 y_{1,0} = h(y_{2,0} + \frac{K_1 y_{2,0}}{2})$		$K_2 y_{2,0} = h(-(y_{1,0} + \frac{K_1 y_{1,0}}{2}))$
	$K_3 y_{1,0} = h(y_{2,0} + K_2 y_{2,0})$		$K_3 y_{2,0} = h(-(y_{1,0} + K_2 y_{1,0}))$
	$y_{1,1} = y_{1,0} + \frac{K_0 y_{1,0}}{6} + \frac{K_1 y_{1,0}}{3} + \frac{K_2 y_{1,0}}{3} + \frac{K_3 y_{1,0}}{6}$		$y_{2,1} = y_{2,0} + \frac{K_0 y_{2,0}}{6} + \frac{K_1 y_{2,0}}{3} + \frac{K_2 y_{2,0}}{3} + \frac{K_3 y_{2,0}}{6}$

There are following mathematical operations for computations: **multiplication**,**addition**, **subtraction** and **divison**. Addition in hardware is typically done in adders, multiplication in hardware can be done by either serial or parallel way.

A very promising way is to apply arithmetic operations in modular arithmetic. As an example a multiplication in modular arithmetics is presented.

MULTIPLICATION IN MODULAR ARITHMETIC

Two mostly used algorithms for computation modular multiplication exploit binary shifts:

- Barrett reduction [9] requires the precomputation of the quantity $\mu = \lfloor b^{2k}/m \rfloor$ (b is base, k is number of bits, m is modulo). The main idea of algorithm is based on relation $\lfloor (x/b^{k-1})(b^{2k}/m)(1/b^{k+1}) \rfloor$, where $x = (A \cdot B)$.

- Montgomery multiplication [10] which is based on transformation to modulo space and back to standard representation.

Note: A new algorithm of modular computation is just uder the development at the Brno University of Technology.

CONCLUSIONS

A simple transformation of an initial problem was presented in the paper. Similar constructions could be created for all elementary functions, such as exp, tg, cotg, ln, sinh, ...
The idea above requires software capable of automatically performing the decomposition of the right-hand sides of ordinary differential equations. This new approach has been implemented in a simulation language TKSL. In fact, the well-known rules of differential and integral calculus have been used.

The idea of a very special program for numerical solution of ODEs based on floating point RNS is presented, too. As a result RNS, exact solution and high speedup factor using parallel procesors could be obtained.

Detailed information will be given during the ICNAAM 2013 conference.

ACKNOWLEDGMENTS

This paper has been elaborated in the framework of the project New creative teams in priorities of scientific research, reg. no. CZ.1.07/2.3.00/30.0055 (CZ.1.05/1.1.00/02.0070), supported by Operational Programme Education for Competitiveness and co-financed by the European Social Fund and the state budget of the Czech Republic. The paper includes the solution results of the Ministry of Education, Youth and Sport research project No. MSM 0021630528 and the international AKTION research project Number 64p13.

REFERENCES

1. J. Kunovský, *Modern Taylor Series Method*, FEI-VUT Brno, 1994, habilitation work.
2. T. V. Vu, “Effecient implementation of Chinese remainder theorem for sign detection and residue decoding,” in *IEEE Transactions on Computers*, IEEE Computer Society, 1985, vol. 34, pp. 646–651.
3. K. M. Elleithy, and M. A. Bayoum, “A $\theta(1)$ algorithm for modulo addition,” in *IEEE Transactions on Circuits and Systems*, IEEE Computer Society, 1990, vol. 37, pp. 628–631.
4. A. Omondi, and B. Premkumar, *Residue Number Systems - Theory and Implementation*, Imperial College Press, London, 2007, pp. 193–195, ISBN 1-86094-866-9.
5. L. P. G.A. Orton, and S. E. Tavares, “New Fault tolerant techniques for residue number systems,” in *IEEE Transactions on Computers*, IEEE Computer Society, 1992, vol. 41, pp. 1453–1464.
6. A. Nannarelli, and G. C. Cardarilli, “Reducing Power Dissipation in FIR Filters using the Residue Number System,” in *IEEE Midwest Symposium on Circuits and Systems*, Institute of Electrical and Electronics Engineers, 2001, vol. 2, pp. 305–308.
7. A. S. R. L. Rivest, and L. Adleman, “A method for obtaining digital signatures and public-key cryptosystems,” in *Communications of the ACM*, ACM, 1978, vol. 2, pp. 120–126.
8. K. Mikulášek, *Polynomial Transformations of Systems of Differential Equations and Their Applications*, Ph.D. thesis, FEI Brno University of Technology (2000).
9. P. Barrett, “Implementing the Rivest Shamir and Adleman Public Key Encryption Algorithm on a Standard Digital Signal Processor,” in *Advances in Cryptology 86*, Springer Berlin Heidelberg, 1987, vol. 263, pp. 311–323.
10. P. l. Montgomery, “Modular Multiplication Without Trial Division,” in *Matematics of Computation*, American Mathematical Society, 1985, vol. 44, pp. 519–521.
11. E. Hairer, S. P. Nørsett, and G. Wanner, *Solving Ordinary Differential Equations I*, vol. Nonstiff Problems. Springer-Verlag Berlin Heidelberg, 1987, ISBN 3-540-56670-8.
12. E. Hairer, and G. Wanner, *Solving Ordinary Differential Equations II*, second revised ed. with 137 Figures, vol. Stiff and Differential-Algebraic Problems. Springer-Verlag Berlin Heidelberg, 2002, ISBN 3-540-60452-9.