Extended precision data types for the development of the original computer aided engineering applications

A Pescaru¹, E Oanta², T Axinte³ and A-D Dascalescu³
¹Constanta Maritime University, Faculty of Navigation and Naval Transport, 104 Mircea cel Batran Street, 900663, Constanta, Romania
²Constanta Maritime University, Faculty of Naval Electro-Mechanics, 104 Mircea cel Batran Street, 900663, Constanta, Romania
³Politehnica University of Bucharest, Faculty of Power Engineering, 313 Splaiul Independentei, Sector 6, 060042, Bucharest, Romania

E-mail: eoanta@yahoo.com

Abstract. Computer aided engineering is based on models of the phenomena which are expressed as algorithms. The implementations of the algorithms are usually software applications which are processing a large volume of numerical data, regardless the size of the input data. In this way, the finite element method applications used to have an input data generator which was creating the entire volume of geometrical data, starting from the initial geometrical information and the parameters stored in the input data file. Moreover, there were several data processing stages, such as: renumbering of the nodes meant to minimize the size of the band length of the system of equations to be solved, computation of the equivalent nodal forces, computation of the element stiffness matrix, assemblation of system of equations, solving the system of equations, computation of the secondary variables. The modern software application use pre-processing and post-processing programs to easily handle the information. Beside this example, CAE applications use various stages of complex computation, being very interesting the accuracy of the final results. Along time, the development of CAE applications was a constant concern of the authors and the accuracy of the results was a very important target. The paper presents the various computing techniques which were imagined and implemented in the resulting applications: finite element method programs, finite difference element method programs, applied general numerical methods applications, data generators, graphical applications, experimental data reduction programs. In this context, the use of the extended precision data types was one of the solutions, the limitations being imposed by the size of the memory which may be allocated. To avoid the memory-related problems the data was stored in files. To minimize the execution time, part of the file was accessed using the dynamic memory allocation facilities. One of the most important consequences of the paper is the design of a library which includes the optimized solutions previously tested, that may be used for the easily development of CAE cross-platform applications. Last but not least, beside the generality of the data type solutions, there is targeted the development of a software library which may be used for the easily development of node-based CAE applications, each node having several known or unknown parameters, the system of equations being automatically generated and solved.
1. Introduction
The definition of the most common data types as user defined distinct types is useful for the definition of the upper level sophisticated data types, as well as for the readily customization of a software library which may be included in a new application and that has specific requirements. Moreover, if all the programs use such definition it is easier to include these components in an upper level software, i.e. to integrate the components of a hybrid engineering model. The flexible definition of the basic data types is a long run concern of the authors and there are several aspects to be considered when a given type is conceived.

2. Criteria regarding the definition of flexible data types
When a major software project was started years ago, a basic concern was about the limited capacity of the memory where large arrays had to be stored. In order to solve this problem there was decided that the best choice was to store the data on the hard disk, as a random access file specific to a given programming language. Because various programming languages offered different data processing facilities, it was developed an interface based on constant structure text files. Moreover, various original software applications developed in different programming languages could be connected using this new interface, being possible to create multi-language applications. Improving this data processing method was a constant concern of the authors, [1, 2, 3].

An important objective was to decrease the access time, i.e. the data processing time. Several solutions were conceived and implemented: optimization of the memory use by defining a vector of double connected circular lists, creation of a RAM disk where the most frequently used files are stored, [4].

Another direction was to develop some libraries in order to offer a facile access to all the operations with the data types previously defined. Using this platform there were rapidly developed turnkey applications in various fields, [5], most of them solving numerical programming problems: analytic models in applied elasticity, physics, electro-technics; numerical methods; FEM and FDM applications; FEM input data generators; FEM output data processors; experimental data processors; visualization software instruments; hybrid models.

![Figure 1. Data types, circular double connected list and interface between the programming languages.](image_url)
An important issue regarding the computer based models and in numerical programming is the accuracy of the results, [6]. There are several methods to improve the accuracy of the results, being very important to have a precise data type of the variables used to perform the calculi. For the example in figure 1, the ‘real’ type is designated as ‘MATRIX_VALUE’, while the ‘integer’ type used as an index is ‘INDEX_TYPE’. If the real variables of the application to be developed are in a common range of values, there may be selected a simple precision type by modifying the ‘MATRIX_VALUE’ type. In this case it is possible to load a larger amount of data in the memory and the application to be faster. If accuracy is important, the ‘MATRIX_VALUE’ type may be defined to become an extended precision real type. All the upper level libraries and software applications are using this type, in this way being possible to modify the entire software application to be developed by performing minimum changes of the code.

However, there are special cases when a special approach is needed.

3. Discussions

There are certain cases when the numerical calculus results may become non-predictable.

Let us consider the calculus of the coefficients $\alpha$ and $\alpha'$, [7], (3), figure 1. The general forms of these coefficients are:

$$\alpha = -\chi \cdot \frac{\pi^2}{8} \sum_{n=1,3,5,...}^{\infty} \frac{1}{n^2} \left\{ (-1)^{n-1} \left[ \frac{\sin\left(\frac{\pi \cdot n \cdot z}{b}\right)}{\cosh\left(\frac{\pi \cdot n \cdot z}{2 \cdot b}\right)} \right] \cdot \cos\left(\frac{\pi \cdot n \cdot y}{b}\right) \right\},$$

$$\alpha' = -\chi \cdot \frac{\pi^2}{8} \sum_{n=1,3,5,...}^{\infty} \frac{1}{n^2} \left\{ (-1)^{n-1} \left[ \frac{\cosh\left(\frac{\pi \cdot n \cdot z}{b}\right)}{\cosh\left(\frac{\pi \cdot n \cdot z}{2 \cdot b}\right)} \right] \cdot \sin\left(\frac{\pi \cdot n \cdot y}{b}\right) \right\},$$

where

$$\chi = \frac{1}{3} \left\{ 1 - \frac{192}{\pi^5} \cdot \frac{b}{h} \cdot \sum_{n=1,3,5,...}^{\infty} \frac{1}{n^3} \cdot \frac{\sin\left(\frac{n \cdot \pi \cdot b}{2 \cdot b}\right)}{\sin\left(\frac{n \cdot \pi \cdot z}{b}\right)} \right\}$$

and

$$\begin{cases} \tau_{x'} = \frac{M_x}{\alpha \cdot h \cdot b^2} \\ \tau_{y'} = \frac{M_x}{\alpha' \cdot h \cdot b^2} \end{cases}$$

The number of terms in the above sums must be carefully chosen because there may be generated large numbers which can lead to large round-off errors.

Same problem may be found in the calculus of the integral expressed as a summation, [8], (8):
In this way, there are extreme cases when, for some shapes of the calculus domain, the previous integral had different results if it was computed using a numerical method or if it was computed using (5). Moreover, in these extreme cases the same application used to compute the results of relation (5) offered different values on different computers or using different compilers.

It results that the calculus of the series requires a special attention in order to have an appropriate accuracy of the results. A solution is to design a user defined type having an extended precision. Moreover, this solution should be flexible in order to have a customized precision with respect to the problem to be solved.

From among many libraries that could solve extended precision and accuracy problems, two are to be noted for their versatility and support from the vast community that contribute to their development:

1. GNU MP - The Multiple Precision Arithmetic Library
2. GNU MPFR - The Multiple Precision Floating-Point Reliable Library

GNU MP, also designated as GMP, is a flexible library developed in C. It offers customizable precision for the basic data types: integer type and floating-point types. The main strength of this solution consists in the highest speed arithmetic for the applications which require more precision than the standard C types.

Many applications require a precision of some hundreds of bits. However, there are situations when thousands or millions of bits are required. This arbitrary precision solution offers high performance in both cases. In this way, the size of the operands leads to the selection of a given algorithm and meanwhile by minimizing the overhead.

Some of the most important solutions to maximize the speed are: defining basic arithmetic data types as full words, creating sophisticated algorithms, using high speed assembly code in the most often used loops and defining the speed as the most important target, consequently less attention being dedicated to the minimization of the complexity or to obtain a higher degree of refinement.

The library contains various classes of high-level types arithmetic functions:

• ‘mpz’ functions operate with high-level signed integers (≈150 arithmetic and logic functions).
• ‘mpq’ functions operate with high-level rational, being included almost 35 functions. The ‘mpz’ functions may be also utilized if they are separately used for the numerator, respectively for the denominator.
• ‘mpf’ functions operate with high-level floating-point (≈ 35 functions). These functions are useful if the ‘double’ type is not accurate enough.
• Classes of C++ functions which are interfaces to the previously mentioned functions.

Up-to-date information on GMP may be found in reference [9]. GMP is used in several computer algebra systems and also in ‘Computational Geometry Algorithms Library’, [10], where the use of the simple using ordinary floating point CPU math is not accurate enough and it even may lead to the so called ‘explosion’ of the geometry algorithms.

On the other hand, the main characteristics of MPFR, [11, 12], which makes the difference with respect to most customizable precision floating-point solutions, are:
• The results of the codes which use MPFR do not depend on the size of the word for a given machine (the mp_bits_per_limb is 64 on many processors), this means portability.
• High degree of customization of the precision, being possible to set the number of bits individually for each variable.
• Apart from the IEEE 754-1985 standard, the library includes four modes for the round-off operations for both simple calculi and for mathematical functions.

There may be noticed that the precision of the double-precision floating-point variables may be reached in MPFR using a 53 bits precision for the basic arithmetic operations and the square root.

The basic idea of the MPFR approach is that the operation is computed using the so called ‘infinite accuracy’ and, using the previously selected rounding mode, the result is rounded to the precision defined for the destination variable. The functions for floating-point operations are smoothly extending the IEEE 754 arithmetic. In this way the same result may be obtained on any computer, without any dependence on the size of the word on the current machine, the compiler type of the operating system.

Moreover, if two variables of lower precision are multiplied and the result is a variable having a large precision, the calculus being done using the previously mentioned ‘infinite accuracy’, it results a full precision number. This is an example regarding the fact that MPFR does not verify the accuracy during the computations. Interesting ideas regarding the loss of precision reasons and the use of the MPFR library to increase the accuracy are given in reference [13].

4. Conclusions
Using extended precision data representation was an important concern of the worldwide researchers. Using higher internal precision, there were created simpler, faster and more accurate algorithms for linear algebra problems [14, 15]. At present new ideas of research regarding the enhancement of the precision are followed, being created new algorithms for normalizing, dividing and square root calculus of the floating point expansions, [16].

In computer based engineering modelling there are certain cases, for instance the calculus of the series, when the necessary precision may be acquired only if extended precision types are used. Thus, in series (1), (2) and (5) there are subtracted two large numbers which might be nearly equal, this operation being known as catastrophic cancellation. Thus, the calculation may become dominated by errors. This loss of significance may be prevented using various mathematical particular methods or using scale factors. An IT general solution is to use arbitrary precision basic types which have a precision higher than the basic types offered by a commonly used programming language. So far, there were studied two solutions. Thus, GMP and MPFR offer important facilities regarding the use of the arbitrary precision types: flexible solutions, possibility to develop cross-platform applications, GNU license. A software project based on extended precision data types is under development. Special attention will be dedicated to the definition of some flexible data types and to the development of the according functions, all these being included in the basic libraries. After a thorough testing of the new basic types, the next stage will be the development of dedicated libraries which solve problems regarding: numerical methods and geometry data processing. The topmost level is dedicated to the development of turnkey applications for the computer based models.

Acknowledgement
Concepts Ideas regarding the computer based instruments in applied elasticity are the result of the models developed in the framework of the MIEC2010 bilateral Ro-Md research project, E. Oanta, C. Panait, L. Lepadatu, R. Tamas, M. Constantinescu, I. Odagescu, I. Tamas, G. Batrinca, C. Nistor, V. Marina, G. Iliadi, V. Sontea, V. Marina, V. Balan, V. (2010-2012), “Mathematical Models for Inter-Domain Approaches with Applications in Engineering and Economy”, MIEC2010 - Bilateral Romania-Moldavia Scientific Research Project, under the supervision of the National Authority for Scientific Research (ANCS), Romania, that is the follow-up of the ID1223 scientific research project: E. Oanta, C. Panait, B. Nicolescu, S. Dinu, A. Pescaru, A. Nita, G. Gavrila, (2007-2010), "Computer Aided Advanced Studies in Applied Elasticity from an Interdisciplinary Perspective", under the supervision of the National University Research Council (CNCSIS), Romania.
References

[1] Kumbetlian G, Bocanete P, Oanta E, Fit M and Magardician N Matrices Described as Files. Pascal library. Interfaces with GWBasic, BasicCAD, C, AutoLISP languages. Demo Programs. Dedicated editor The 2nd National Conference on boundary and finite element (University of Sibiu) 13/05/93-15/05/93 pp 37-43

[2] Oanta E 'Matrix' Variable Type Improvements and Applications, The 3rd National Conference on boundary and finite (Merchant Marine Institute: Constanta) 25/05/95-27/05/95 pp 44-49

[3] Oanta E, Nicolescu B and Versatile A 1999 ASME Design Engineering Technical Conference 1999 Proceedings of DETC99 (Las Vegas: Nevada)

[4] Oanta E 1993 Methods to optimize the automatic solving of the numerical calculus problems Annual Scientific Session ICES and Naval Equipment pp 33-34

[5] Oanta E 1998 Advanced software for the data processing in the computational mechanics domain International Symposium on Marine Technologies and Management (Ovidius University of Constanta) II pp 151-158

[6] Oanta E 2012 Approximations in structural analytical studies Constanta Maritime University Annals 18 pp 129-134

[7] Oanta E, Panait C, Lazaroiu G and Dascalescu A E 2014 Computer aided instrument to be used as an automatic design component Advanced Materials Research 1036 pp 1017-1022

[8] Oanta E, Constantinescu E, Raicu A and Axinte T 2014 Analytic General Solution Employed to Calculate the Geometrical Characteristics in Structural Problems Advanced Materials Research 1036 pp 697-702

[9] https://gmplib.org/, Accessed: 05/01/2015

[10] http://www.cgal.org/, Accessed: 05/01/2015

[11] http://www.mpfr.org/, Accessed: 05/01/2015

[12] Fousse L, Hanrot G, Lefèvre V, Pélissier P and Zimmermann P 2007 A multiple-precision binary floating-point library with correct rounding ACM Transactions on Mathematical Software 33(2)

[13] Ghazi K R, Lefèvre V, Théveny P and Zimmermann P 2010 Why and how to use arbitrary precision Computing in Science & Engineering 12(3) p 5

[14] Li X, Demmel J, Bailey D, Henry G, Hida Y, Iskandar J, Kahan W, Kang S, Kapur A, Martin M, Thompson B, Tung T and Yoo D 2002 Design, Implementation and testing of Extended and Mixed Precision BLAS ACM Transactions on Mathematical Software 28(2) pp 152-205

[15] Krougly Z L and Jeffrey D J Implementation and application of extended precision in Matlab Mathematical Methods and applied Computing, pp 103-108

[16] Joldes M, Marty O, Muller J and Popescu V 2015 Arithmetic algorithms for extended precision using floating-point expansions LIP - ENS Lyon, LAAS-CNRS, ENS Cachan (Research Report)