Computing and Informatics, Vol. 28, 2009, 115–125

DEVELOPMENT AND OPTIMIZATION OF COMPUTATIONAL CHEMISTRY ALGORITHMS

Grzegorz MAZUR, Marcin MAKOWSKI

Faculty of Chemistry
Jagiellonian University
Ingardena 3, 30-060 Kraków, Poland
e-mail: {mazur, makowskm}@chemia.uj.edu.pl

Revised manuscript received 29 June 2008

Abstract. The challenges specific to the development of computational chemistry software are discussed. Selected solutions are presented, including examples of algorithmic optimizations and improved load-balancing for parallel calculations. A software framework for development of new quantum-chemical algorithms is proposed. Key design points are discussed. Optimization techniques are briefly described. Important implementation aspects, like automatic code generation, are highlighted.

Keywords: Computational chemistry, optimization, parallelization, software framework

1 INTRODUCTION

Significantly growing demand for higher quality computational results for large molecular systems can be recently observed. This trend is accompanied by advances in theoretical chemistry resulting in proliferation of quantum-chemical models. As a result, the needs of computational chemistry community are constantly rising. Efficient algorithms and domain-specific tools facilitating software development are necessary to fulfil them.

The aim of the paper is to present selected challenges faced by developers of computational chemistry software, and to show how they can be solved. The paper is structured as follows. In Section 2 the overall design of the software framework is presented. In the next section selected algorithmic optimizations are described.

Code generation methods are presented in Section 4. Selected optimization techniques are briefly described in Section 5. A short discussion of the parallelization is presented in Section 6, and conclusions are drawn in Section 7.

2 DESIGN

Niedoida [1] is a general-purpose computational chemistry package, built as a set of libraries implementing quantum-chemical and microelectrostatic calculations. Main Niedoida's design objectives are clear modular structure and extensibility to facilitate its usage as a development platform [2, 3, 4] in the field of computational chemistry.

The object-oriented design is used to achieve loose coupling. Generic programming techniques are applied to increase efficiency. Extensive use of generic algorithms and advanced memory management simplifies the implementation of modern algorithms and methods of computational chemistry. These features are rather untypical for existing quantum-chemistry packages.

In the case of a typical quantum-chemical calculations, the state of a program during its exection is quite complex. Maintaining the state and then passing parts of it to the next stages of calculations puts a significant burden on the programmer. Identification of the key abstractions and proper encapsulation significantly alleviates the problem. For these reasons, the object-oriented approach is very well suited to the problem at hand.

As an example of such approach let us consider calculation of the two-electron repulsion integrals. This is one of the key abstractions in the majority of the quantum-chemical calculations. It is represented as the TwoElectronIntegralEngine class, which defines minimal interface necessary to perform the calculations. The derived classes implement specific methods of calculating the values of the integrals and encapsulate the state required to perform the calculations. Aggregation of the derived classes allows one to build an efficient computational scheme, reusing the already implemented functionality.

Niedoida is implemented in C++. The choice of the implementation language allows one to obtain both clear expression of the artifacts from the problem domain and efficient implementation of the computationally demanding kernels. The key algorithms are parallelized according to the Single Program Multiple Data paradigm. Message Passing Interface (MPI) is used as the implementation framework.

Niedoida features a wide range of standard computational chemistry methods, including Hartree-Fock [5], Kohn-Sham [6], Moller-Pleset second order perturbation theory [5], Configuration Interaction Singles [5], Time-Dependent Density-Functional Theory [7] and Self Consistent Polarization Field [8]. Each of them can be used either directly or as building block for new methods.

3 ALGORITHMIC OPTIMIZATION EXAMPLES

The resources necessary to perform calculations primarily depend on the level of approximation used. However, within a given level of approximation, the efficiency may vary significantly depending on the algorithm. In the next sections two examples of algorithmic optimizations developed in Niedoida are presented.

3.1 Two-Electron Integrals Engine

The two-electron repulsion integrals (ERI) are defined as

$$(ij|kl) = \iint \mathrm{d}\boldsymbol{r}_1 \mathrm{d}\boldsymbol{r}_2 \chi_i(\boldsymbol{r}_1)\chi_j(\boldsymbol{r}_1)\frac{1}{|\boldsymbol{r}_1 - \boldsymbol{r}_2|}\chi_k(\boldsymbol{r}_2)\chi_l(\boldsymbol{r}_2) \tag{1}$$

where χ denotes an atomic orbital. Calculation of the two-electron integrals consitutes the most significant part of the Hartree-Fock and hybrid DFT calculations. In most cases the calculations are performed using the Gaussian Type Orbitals (GTO). A GTO is a linear combination (contraction) of normalized Gaussian functions (primitives), augmented with spherical harmonics representing the directional properties of the orbital.

One of the most popular schemes of calculating ERIs [9] starts from the set of Boys functions [10] F_N. They are used to calculate auxiliary s-type integrals $(ss|ss)^N$

$$\Theta^N_{000000000000} = (ss|ss)^N = \frac{2\pi^{\frac{5}{2}}}{pq\sqrt{p+q}} K^{xyz}_{ab} K^{xyz}_{cd} F_N(\alpha R^2_{PQ}), \tag{2}$$

where for primitive Gaussian functions with exponents equal a, b, c, d and centered on A, B, C, D respectively, relevant quantities are defined as follows

$$p = a + b \qquad q = c + d \qquad \alpha = \frac{pq}{p+q}$$

$$P_x = \frac{aA_x + bB_x}{a+b} \qquad X_{PQ} = P_x - Q_x \qquad K^{xyz}_{ab} = exp(-\frac{ab}{a+b}R^2_{AB}). \tag{3}$$

S-type auxilliary integrals are further transformed, using a chain of recursions and performing contraction, into final integrals over contracted Cartesian basis functions:

$$\Theta^N_{000000000000} \rightarrow \Theta_{i_x i_y i_z j_x j_y j_z k_x k_y k_z l_x l_y l_z} \rightarrow (a_{i_x i_y i_z} b_{j_x j_y j_z} | c_{k_x k_y k_z} d_{l_x l_y l_z}). \tag{4}$$

The recursion steps include *vertical recursion* [11] that moves angular momentum from $(ss|ss)^N$ integrals into the first function of the first electron producing $(xs|ss)$ integrals, *electron transfer* [12] that allows to transfer it from one to the other electron generating $(xs|ys)$ integrals and *horizontal recursion* [11] used to move it from one function of the given electron to the other one.

In actual calculations the integrals over the whole shells of basis functions sharing the same angular momentum are evaluated simultaneously to avoid recalculation of the common intermediate auxiliary integrals. Usually, there is also more than one way to reach a specific integral by means of recurrence relations. These two factors combined together create a complicated optimization problem of creating an algorithm that would minimize the number of operations required to evaluate all the integrals for a given shell-quartet type. The best solution of this problem can be in principle obtained using exhaustive search in the space of possible solutions. However, due to the size of the scanned space such an approach is computationally prohibitive in most of the interesting cases. Instead we can use some heuristic techniques [2] that allow us to find near-optimal solutions and generate the algorithms. It turns out that the algorithms are very competitive in terms of efficiency to those proposed earlier in the literature [12, 13].

To illustrate the methodology let us consider a relatively simple application to the vertical recursion step. The generation of the quasi-optimal algorithm starts from the final auxiliary integrals required for a specific shell-quartet type. In case the shell quartet with total angular momentum equals L this set comprises all the possible $\Theta^0_{i_x i_y i_z 000000000}$, where $0 < i_x + i_y + i_z \leq L$. For each such quantity the Cartesian direction of the generating recursion is chosen. It corresponds to the smallest non-zero component from the set of (i_x, i_y, i_z) describing the final quantity. If two or more angular components are equal, the lexicographically last of them is selected. In the next step the individual recursions are inspected and all new auxiliary integrals different than $\Theta^N_{000000000000}$ are added to the working set. These two steps are repeated for all the yet unprocessed quantities in the current working set. When the pool of such becomes empty, the working set is ordered by the total angular momentum and individual recursions in the previously chosen directions are generated.

We performed similar optimizations for all recursion types present in the modified Obara-Saika scheme. It allowed us to generate near-optimal code covering all the shell quartet types up to $(gg|gg)$ [2].

3.2 Self-Consistent Polarization Field

The polarization energy of an excess charge in polarizable medium is the energy arising from the interaction between the charge and the induced dipole moments of the surrounding molecules [14, 15]. Its value is an important parameter in a wide range of models describing excited electronic states, transport effects and electronic conduction, crucial for rationalizing those properties of organic solids [15, 16, 17, 18, 19] which are potentially relevant for the applications of those systems in electronics and optoelectronics. An analogous problem has to be solved in molecular dynamics employing polarizable force fields.

Two alternative methods are available for calculating polarization energy. The Fourier Transform (FT) method [14] makes use of the translational symmetry of the system to express its dielectric response to electric fields of different wavevectors.

By expanding the charge field into its Fourier components, polarization energy is derived in an explicit algebraic form.

The Self-Consistent Polarization Field (SCPF) method [15, 8] does not assume translational symmetry. Therefore, it is applicable to a wider range of systems, including perfect and imperfect bulk crystals and surface regions as well as non-crystalline systems. Moreover, it allows for explicit treatment of relaxation effects.

The method is based on splitting the system under study into two parts. In the inner part, confined to a sphere centered at the excess charge, the molecules are treated as discrete entities and their dipole moment is calculated directly from the field of the charge and the induced field of other molecules. The outer region is described as polarizable continuum.

The self-consistent equations for the dipole moment of molecules located in the inner region may be written as [8]

$$\boldsymbol{\mu}_n = \boldsymbol{\alpha}_n \boldsymbol{E}_n^0 + \boldsymbol{\alpha}_n \sum_{n' \neq n} \boldsymbol{T}_{nn'} \boldsymbol{\mu}_{n'} \tag{5}$$

where $\boldsymbol{\mu}_n$ is the induced dipole moment, $\boldsymbol{\alpha}_n$ stands for polarizablility tensor, $\boldsymbol{E}_n^0$ is the external electric field and

$$\boldsymbol{T}_{nn'} = \frac{1}{4\pi\epsilon_0} \nabla\nabla |\boldsymbol{r}_n - \boldsymbol{r}|^{-1}_{\boldsymbol{r}\to\boldsymbol{r}_{n'}}. \tag{6}$$

is the dipole tensor.

Due to the number of molecules present in typical calculations, solving Equation (5) directly is not feasible and iterative methods have to be used instead. Originally, the Jacobi method was applied to the SCPF equations [8].

However, this particular choice of the solver is not optimal. Firstly, for some of the systems the parameters are beyond the convergence radius of the Jacobi model, and convergence has to be enforced using e.g. damping. Additionally, even when the Jacobi method converges, the convergence rate is usually poor. Both of the drawbacks are inherent to the Jacobi method.

Assuming nonsingularity of the polarizability tensors, which holds for all physically conceivable systems, Equation (5) may be transformed to [3]

$$\boldsymbol{A}\boldsymbol{\mu} = \boldsymbol{E}_0 \tag{7}$$

where

$$\boldsymbol{A} = \begin{pmatrix} \boldsymbol{\alpha}_1^{-1} & -\boldsymbol{T}_{12} & \dots & -\boldsymbol{T}_{1n} \\ -\boldsymbol{T}_{21} & \boldsymbol{\alpha}_2^{-1} & & \\ \vdots & & \ddots & \vdots \\ -\boldsymbol{T}_{n1} & & \dots & \boldsymbol{\alpha}_n^{-1} \end{pmatrix}. \tag{8}$$

The reason to perform the transformation is the fact that Equation (7) is symmetric. This is due to the permutational symmetry of the dipole tensors, $\boldsymbol{T}_{ij} = \boldsymbol{T}_{ji}$, and to the symmetry of the polarizability tensors. In addition, this formulation

of the SCPF equations shows that $\boldsymbol{A}$ may be regarded as the inverse of the polarizability as modified by the dipolar interactions. Therefore, it must be positive definite.

The properties of the transformed SCPF equations allow for application of the Conjugate Gradients (CG) method. This leads to much better convergence of the iterative process and, as a result, significantly increased efficiency [3].

4 CODE GENERATION

The complexity of computational kernels in quantum-chemical calculation calls for methods of describing the formulas and algorithms in a high-level, easy to understand and maintain way instead of relatively low-level description possible in typical, general-purpose languages. While there are some attempts to employ automatic code generation in the computational chemistry software [20, 21], the technique is not very often used in typical quantum-chemical programs. In the next sections two different approaches to the code generation implemented in niedoida are presented.

4.1 Two-Electron Integrals Engine

The optimized code generation procedure as applied to the horizontal recursion stage of the ERI calculation consists of a few algorithmic steps. First, we find all needed intermediate classes of integrals for the final required shell-quartet type. Then we generate the actual recurrence relations for all the members of all classes using similar rules as in Section 3.1 to choose the Cartesian direction in which the transfer takes place. Together with this step all unreferenced intermediates are detected and eliminated recursively. In the third stage we proceed in a bottom-up fashion through a set of equations and replace all intermediates that are used only once with appropriate generating relations. Finally, simple algebraic manipulations are performed to evaluate polynomial expressions on the right hand side in a way that provides the lowest possible number of FLOPs and MOPs.

The procedure described above can be generalized to any type of the required shell-quartet. It was implemented for the general case as a relatively short C++ program generating code providing the final optimized set of the recurrence relations for the use in the integral engine of Niedoida.

4.2 Exchange-Correlation Functionals

The exchange-correlation functional is the central notion of the Density Functional Theory (DFT) [6], an approach to quantum-chemical calculations very popular recently for its competitive speed and accuracy. While the existence of the universal functional has been proved, its exact formulation remains unknown. This leads to proliferation of approximations, each having specific strong and weak points. The structure of DFT calculations requires several derivatives of the functional to be

computed. The sheer number of approximate functionals, their complexity and the number of the required derivatives poses a significant implementation challenge.

Automatic code generation of the exchange-correlation derivatives in Niedoida is implemented using Yacas [22] computer algebra system program. The functionals are implemented as Yacas functions. Their derivatives are calculated symbolically using the built-in derivative operator. The resulting formulas are simplified using the Common Subexpression Elimination (CSE) algorithm, implemented within Yacas. Performing CSE at this stage using a customized algorithm is necessary because of the length of the formulas, typically exceeding hundred thousand of characters, and their complexity. A straightforward generation procedure would create code which cannot be efficiently compiled by the existing compilers. Finally, the simplified expressions are transformed to C++ using the Yacas built-in code generation functionality.

5 SELECTED OPTIMIZATION TECHNIQUES

The object oriented design of niedoida imposes overhead of the virtual function calls. In the most efficiency sensitive parts of the code the overhead is mitigated by grouping calls into larger batches. Another significant factor influencing the efficiency of the code is the dynamic memory allocation. Its impact is reduced by preallocating the memory before entering the computational kernels. The abstraction penalty is reduced by function inlining and switching to raw memory access via pointers within inner loops. Additionally, code generation and template metaprogramming is used to unroll tight loops.

All the optimization techniques were studied using execution time profiles generated by valgrind [23], and proved crucial for the efficiency of the computational chemistry code.

6 PARALLELIZATION

Most of the existing procedures used for commonly executed quantum-chemical calculations are parallelized with the Single Program Multiple Data (SPMD) model. This approach is in general case not optimal from the efficiency point of view. It is, however, justified by its conceptual simplicity and near-optimal performance for typical applications. The SPMD model requires a scheduler to partition the problem between computational nodes. In this work we propose an efficient and flexible scheduling algorithm. Its key design feature is near-optimal run-time adaptivity of the scheduling process achieved with negligible computational overhead.

The proposed adaptive load-balancing algorithm works as follows. The computational problem is divided into tasks of different sizes. The tasks are stored in a task queue. A node gets the next task from the queue as soon as it completes the previous one. The splitting of the problem into tasks is organized as follows. A fraction $1/f$ of the original problem is divided into n tasks, where f is the split-

ting factor and n stands for the number of nodes. Then the procedure is repeated recursively for the remaining part of the problem. The recurrence is stopped when the size of the remaining part is smaller than the threshold t. Then, the rest is split uniformly into n tasks.

The algorithm is primarily parametrized by the value of f. The larger the value of f, the more fine-grained partitioning is achieved. This results in better balancing, at the cost of increased communication overhead. The optimal value of f is a trade-off between expected non-uniformity of the environment and estimated communication cost. The meaning of the t parameter is more technical. Its value should be decreased when the value of f is growing, to avoid worsening the balancing in the last stage of calculations. After preliminary tests we decided to use $f = n + 1$ and $t = 500n$. These values were used for the benchmark calculations presented below.

As the baseline for performance analysis a simple static load-balancing algorithm was chosen. It splits the computational problem into equally sized tasks which are uniformly distributed between nodes.

Test calculations were performed for sexithiophene at the Hartree-Fock level of theory using the 6-31G** basis. The size of the model system is best characterized by the number of orbitals. In this case the number of orbitals is 514, which is the magnitude typical for commonly performed quantum-chemical calculations. The computations were performed on an SGI Altix 3700 system equipped with 128 Itanium 2 processors (1.5 GHz) and 256 GB RAM. The scaling of the proposed load-balancing algorithm with the number of CPUs is shown in Figure 1.

The relative time required to perform calculations using n CPUs may be described by

$$T_n = \frac{\alpha}{n^\beta} + \gamma$$

where α represents the fraction of the parallelized part and γ stands for the (effectively) serial fraction of the calculations. β describes the deviation from linear speedup. The scaling function described above was fitted to the datapoints obtained for the model system. The fitted parameter values are presented in Table 1.

Algorithm	α	β	γ
Static	0.92	0.69	0.07
Dynamic	0.96	1.00	0.03

Table 1. Fitted parameter values for the load-balancing algorithms

The performed tests confirmed the validity of the key design decisions. A preliminary analysis of the expected hyper-cache effect was also made. The results suggest that it can be achieved only if cache affinity is introduced to the scheduling process. This work is currently underway in our group.

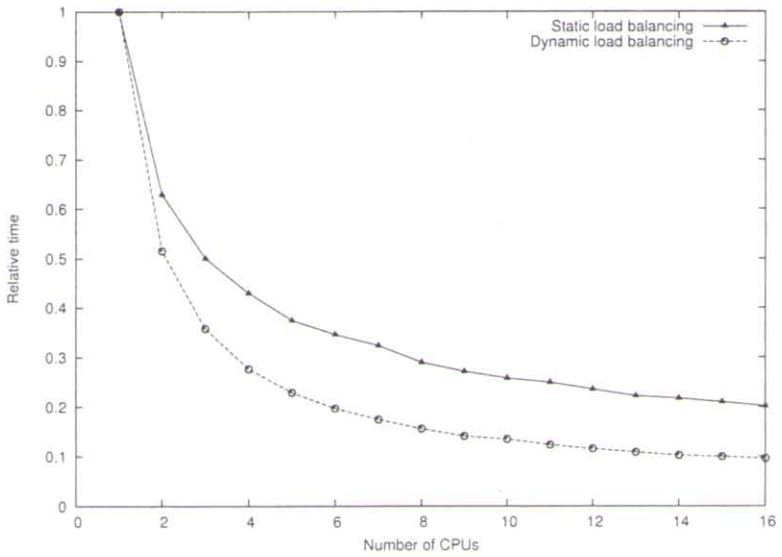

Fig. 1. Time of SCF calculations for the model system in function of number of CPUs. The time is relative to the 1 CPU case.

7 CONCLUSIONS

Object-oriented paradigm is well suited for the construction of computational chemistry software. Abstraction, encapsulation and code reuse proved crucial for managing the complexity of quantum-chemical code. The application of these principles allowed us to build an easy to maintain and extend quantum-chemical software framework. Simple optimization techniques proved sufficient to mitigate the overhead and keep the efficiency at the level not worse than that achieved by more conservative solutions. On this basis we have managed to implement advanced domain-specific algorithms with a relatively little effort.

Additionally, the application of standard software engineering methods, like automatic code generation, simplified the code, shortened the development time and reduced the testing and debugging workload.

Acknowledgments

Support from ACK Cyfronet (grant MEiN/SGI3700/UJ/077/2006) is acknowledged.

REFERENCES

[1] Mazur, G.—Makowski, M.—Piskorz, W.—Ćwiklik, L.—Sterzel, M.—Radoń,M.—Jagoda-Ćwiklik, B.—Kulig, W.—Błażewicz, D.: Niedoida 0.3., 2007.

[2] Makowski, M.: Simple Yet Powerful Techniques for Optimization of Horizontal Recursion Steps in Gaussian-Type Two-Electron Integral Evaluation Algorithms. Int. J. Quantum Chem., Vol. 107, 2007, p. 30.

[3] Mazur, G.: An Improved SCPF Scheme for Polarization Energy Calculations. J. Comp. Chem., Vol. 29, 2008, p. 988.

[4] Mazur, G.—Wlodarczyk, R.: Application of the Dressed Time Dependent Density Functional Theory for the Excited States of Linear Polyenes. Submitted 2008.

[5] Helgaker, T.—Jorgensen, P.—Olsen, J.: Molecular Electronic Structure Theory. John Wiley and Sons Ltd., 2000.

[6] Kohn, W.—Sham, L. J.: Self-Consistent Equations Including Exchange and Correlation Effects. Phys. Rev. A, Vol. 140, 1965, p. 1133.

[7] Casida, M.: Time-Dependent Density Functional Response Theory of Molecular systems: Theory, Computational Methods, and Functionals. In: J. Seminario, ed., Recent developments and applications in density functional theory. Elsevier, Amsterdam 1996.

[8] Knowles, D. B.—Munn, R. W.: Polarization Energy Calculations in Molecular Crystals. J. Mat. Sci., Vol. 5, 1994, p. 89.

[9] Obara, S.—Saika, A.: Efficient Recursive Computation of Molecular Integrals over Cartesian Gaussian Functions. J. Chem. Phys., Vol. 84, 1986, p. 3963.

[10] Boys, S. F.: Proc. Roy. Soc. A, Vol. 200, 1950, p. 542.

[11] Head-Gordon, M.—Pople, J. A.: A Method for Two-Electron Gaussian Integral and Integral Derivative Evaluation Using Recurrence Relations. J. Chem. Phys., Vol. 89, 1988, No. 9, p. 5777.

[12] Lindh, R.—Ryu, U. et al.: The Reduced Multiplication Scheme of the Rys Quadrature and New Recurrence Relations for Auxiliary Function Based Two-Electron Integral Evaluation. J. Chem. Phys., Vol. 95, 1991, p. 5889.

[13] Johnson, B. G.—Gill, P. M. W. et al.: The Efficient Transformation of (m0n0) to (abcd) Two-Electron Repulsion Integrals. Chem. Phys. Lett., Vol. 206, 1993, p. 229.

[14] Bounds, P. J.—Munn, R. W.: Polarization Energy of a Localized Charge in a Molecular Crystal. Chem. Phys., Vol. 44, 1979, p. 103.

[15] Silinsh, E. A.: Organic Molecular Crystals: Their Electronic States. Springer, Berlin 1980.

[16] Petelenz, P.—Slawik, M. et al.: Theoretical Calculation of the Electroabsorption Spectra of Polyacene Crystals. J. Chem. Phys., Vol. 105, 1996, No. 11, p. 4427.

[17] Mazur, G.—Petelenz, P. et al.: Theoretical Calculations of the Electroabsorption Spectra of Perylenetetracarboxylic Dianhydride. J. Chem. Phys., Vol. 118, 2003, No. 3, p. 1423.

[18] SCARLE, S.—STERZEL, M. et al.: Monte Carlo Simulation of Li+ Motion in Polyethylene Based on Polarization Energy Calculations and Informed by Data Compression Analysis. J. Chem. Phys., Vol. 123, 2005, No. 15, p. 154909.

[19] EILMES, A.—MUNN, R. W.: Microscopic Calculation of the Energetics of Charged States in Amorphous Polyethylene. J. Chem. Phys., Vol. 120, 2004, No. 16, p. 7779.

[20] CRAWFORD, T. D.—SHERRILL, C. D. et al.: PSI3: An Open-Source Ab Initio Electronic Structure Package. J. Comp. Chem., Vol. 28, 2007, p. 1610.

[21] SALEK, P.—HESSELMANN, A.: A Self-Contained and Portable Density Functional Theory Library for Use in Ab Initio Quantum Chemistry Programs. J. Comp. Chem., Vol. 28, 2007, No. 16, p. 2569.

[22] Yacas: `http://yacas.sourceforge.net/`.

[23] NETHERCOTE, N.—SEWARD, J.: Valgrind: A Framework for Heavyweight Dynamic Binary Instrumentation. In: Proceedings of ACM SIGPLAN 2007 Conference on Programming Language Design and Implementation (PLDI) 2007.

Grzegorz MAZUR is an Assistant Professor at the Department of Computational Methods in Chemistry of Jagiellonian University in Cracow (Poland). He attained his Ph. D. in chemistry in 2001 at the same university. His current research interests include theoretical description of the excited states properties and optimization of quantum chemistry algorithms.

Marcin MAKOWSKI is an Assistant Professor at the Department of Theoretical Chemistry of Jagiellonian University in Cracow (Poland). He attained his M. Sc. degree in 2001 and his Ph. D. in 2004 both in chemistry at the same university, and B. Sc. degree in computer science at University of Mining and Metallurgy in Cracow in 2004. His current research interests include theoretical molecular spectroscopy, linear scaling methods in quantum chemistry and optimization of quantum chemistry algorithms. He participates in several research projects related with the development of theoretical chemistry formalisms and numerical methodologies that allows to calculate efficiently electronic structure of large molecular systems.