Common Format for Quantum Chemistry Interoperability: Q5Cost Format and Library

Anthony Scemama, Antonio Monari, Celestino Angeli, Stefano Borini, Stefano Evangelisti, Elda Rossi

Common Format for Quantum Chemistry Interoperability: Q5Cost format and library

Anthony Scemama [a], Antonio Monari [b], Celestino Angeli [c], Stefano Borini [d], Stefano Evangelisti [a], and Elda Rossi [e]

a) Université de Toulouse et CNRS, Laboratoire de Chimie et Physique Quantiques, 118, Route de Narbonne, F-31062 Toulouse Cedex – France.
b) Dipartimento di Chimica Fisica ed Inorganica, Università di Bologna, Viale Risorgimento 4, I – 40136 Bologna – Italy.
c) Dipartimento di Chimica, Universita` di Ferrara, Via Borsari 46, 44100 Ferrara
d) Center for Biological Sequence Analysis, Technical University of Denmark, Anker Engelunds Vej - Building 301, DK-2800 Kgs. Lyngby, Denmark
e) CINECA, Via Magnanelli 6/3, I-40033 Casalecchio di Reno (BO) – Italy.

Abstract. A particular data format for interchange, specifically designed for Quantum Chemistry programs, was proposed within a COST activity. The new features of the recent version 1.0 are presented, together with the Q5Cost library that allows to use the data format within a user program. The problem of a general description of the wave function is presented as well as the adopted strategy for including it into the data format. Several utilities have been delivered for the use of the developers and the users of the library, as well as various converters for well known Quantum Chemistry codes.

Introduction

The present article describes the design, setup and upgrading of a data format and a FORTRAN library, Q5Cost, for the management of data produced by a generic *ab-initio* Quantum Chemistry (QC) code. The data structure has been defined with the aim of making code interoperability easier in the scientific area of Quantum Chemistry. This activity has been carried out within the COST network, supporting cooperation among researchers in Europe: the former D23 “Metachem” action, now concluded, and the D37 “GridChem” action, started in July 2006 [1].

The first problem to be faced when integrating different QC codes in a common workflow is the lack of a standard and the different data formats adopted by each code. Our suggestion was to adopt such a standard, specifically designed for interchange, and to connect each code in the set through a converter. Of course not to invent “yet another format”, we strongly tried to design a format as general and

flexible as possible and to coordinate ourselves with other similar initiatives in the quantum chemistry context.

The first idea was to adopt two different technologies for describing large binary data and small coded information. HDF5 (Hierarchical Data Format) [2] was the technology of choice for the first type of data, and XML (eXtensible Markup Language) [2] for the second one.

Nevertheless, during the project development, important considerations lead us to the decision to collect all the data structures into a unique data format and, in particular, to design a single user interface for managing all the data. A first definition of the data model and a description of the library has been reported elsewhere [3,4]. In this paper we want to report the state of the art of the library, now at the version 1.0, in addition to a critical examination of the improvements and modifications carried on it.

The next section will discuss the improvements carried on in the data format, in terms of "Geometry", "Basis set" and "Wave function". The same issues are then considered in the following section from the user interface point of view. The strategy adopted for storing the wave function is then discussed in detail, since this is a point that poses difficult problems due to the size and wide variety of possible wave functions. A number of utilities have been set up to facilitate the use of the Q5Cost library and are described in the next session. The utilities are addressed to the library developers but also to the programmers that are using the library in their programs to manage the files produced by a QC calculation. Wrappers are the last topic presented in this paper; they are specific translation programs that use the Q5Cost library for converting the proprietary data format into the Q5Cost data format and vice versa.

The data format

The structure of the first versions of the Q5Cost data format has been fully described elsewhere [3,4], as well as its connections with the inherent HDF5 structure [2]. Here we just want to briefly recall the main features and present in details the modifications introduced with the present version.

Q5Cost is a common interchange format for data coming from *ab initio* quantum chemistry calculations. In this context *ab initio* refers to methods that are based on a wave function rather than an electronic density. In particular, the Density Functional Theory (DFT) methods at the moment are not supported by our format. This choice is justified by the fact that the majority of the codes involved in this project are based on Slater-Determinant expansions of the wave function.

The q5Cost format is flexible and extensible, so we can design it in an incremental way. The data format consists of a collection of chemical objects related within a hierarchical structure in a logical containment relationship as reported in Figure 1.

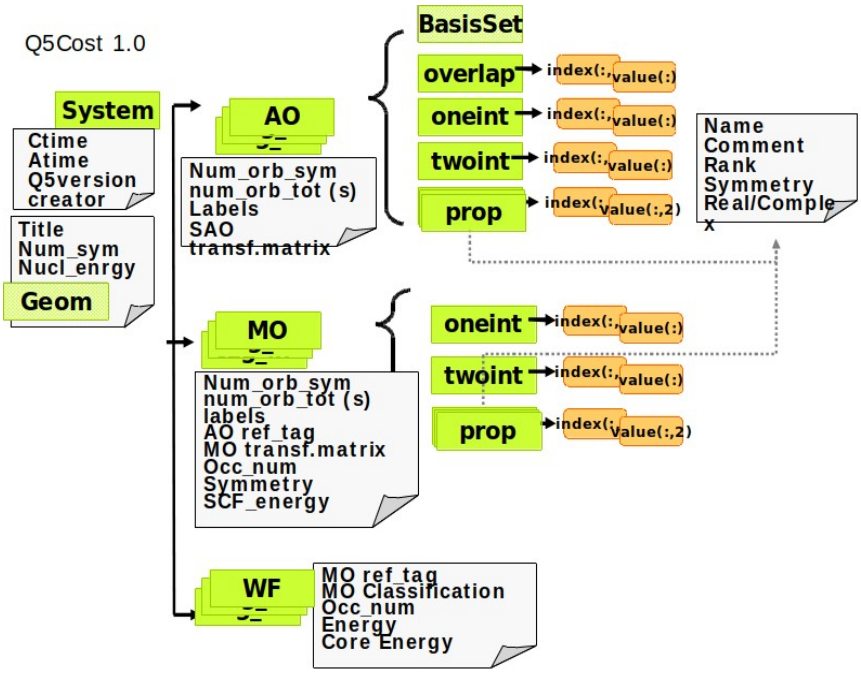

Fig. 1. Schema of the Q5Cost data format

The green boxes are containers (*groups* in HDF5 terminology), and the orange boxes are data (*data sets*), while the metadata (*attributes* describing the data) are listed in the white labels. The light green boxes are the new entities included in the 1.0 version of the data format.

Originally the Q5Cost data format was designed to handle only the large binary data coming from Quantum Chemistry, together with the information (metadata) needed for describing them. The data structures described by the format were basically integrals, on the Atomic and Molecular orbitals, the accompanying metadata entities like electrons number, symmetry indices, nuclear energy, molecular orbital labels, but also coefficients needed to define symmetry adapted orbitals or to transform the Atomic orbitals into Molecular orbitals.

Other small pieces of information, mainly ASCII coded in conventional QC data formats, are needed to define a QC system. They can be classified into three classes: Geometry, Basis set, spatial Symmetry of the system under investigation. In the original version of the data format those data were represented using an XML based language in a separated file [3]. Starting from this version, we decided to merge them into the Q5Cost data format, in order to ensure a unique coherent structure grouping all the specifications on the chemical system. This is a new feature of the format that now includes the whole information about a molecular system.

Data represented into the Q5Cost format belong to two different categories:

1. Data: the large binary quantities used in quantum chemistry for representing integrals, properties and wave functions. They can be stored using matrices with an arbitrary number of indices (rank-n arrays), scale aggressively with the system size, and are normally accessed with a "chunked" approach (i.e., using well-defined blocks of data).
2. Metadata: simple and small pieces of data that describes and better defines the previous data. They represent well-known chemical entities like nuclear energy, molecular orbital labels, and molecular symmetry and can be stored as scalars, vectors or matrices. For example, the nuclear repulsion energy is a floating point scalar, molecular orbitals coefficients are an (N,M) floating point matrix, the associated orbital energies are a floating point vector, the molecular orbital labels are a vector of strings, and so on.

Among the "data" entities we can include two-electron integrals, or atomic orbital overlap, but also other more application-specific objects, like the four particle density matrix. In the modern approaches based on localized orbitals, these matrices are of sparse nature; this encourages the storage of only nonzero elements, each one associated to n indices in the case of a rank-n array. This representation of the data, although not particularly efficient in terms of space occupation, is well-known by the interested parties, easy to debug, and already integrated in the current code-base both for memory representation and file storage of data.

A great part of these data objects can be described by a "generic property" object, provided that we define the matrix rank, the involved operator(s) and basis functions. Nevertheless, since some of these "properties" are well-known chemical entities, and chemists are used to referring to them by name, we provide a specific library access to most of them (overlap, one-electron integrals, two-electron integrals...), in addition to an interface to the "generic property" for handling other properties not explicitly provided by the library. This should ensure both ease of use and general adaptability of the library to either alternative or future theoretical developments.

All the chemical objects described in the Q5Cost data format are related within a hierarchical structure, and logical containment relations can be defined for them. A first (root) container, named System, represents the molecular system as defined by its structural data (chemical composition and spatial geometry). All the general metadata, that cannot be integrated into a specific domain, can be associated to this container. A system can contain several "Domains". The role of the Domain is to group together entities whose indices conceptually refer to the same kind of functions. In the present version three Domains have been defined:

1. Atomic Orbital (AO): refers to the data defined on the AO basis, overlap, one-electron integrals, two-electron integrals and the generic property, i.e. any other property that can be described on the AO basis (dipole moment integrals, for example). This domain contains also the definition of the Basis Set.
2. Molecular Orbital (MO): refers to the data defined on the MO basis, one- and two-electron integrals and the generic property. This domain contains also the transformation matrix needed to define the MO on the AO basis.
3. Wave Function (WF): refers to the definition of the wave function. This will be described if full details in the next section.

4.
One- and two-electron integrals and the generic property are present in both AO and MO domain but they are different quantities. The first set is defined on the AO basis and the second is obtained from the first one thought the transformation matrix, i.e. the matrix containing the molecular orbitals coefficients.
In the data format are present, with respect to the previous version, two additional groups of information about Geometry and Basis set.

Geometry is specified by giving the Cartesian coordinates, the atomic number and label of each atom unique by symmetry. Only a single geometry can be stored, at present, in a given Q5Cost file and this group is contained in the System container. We are considering the possibility to store multiple geometries in a single file, in order to facilitate the treatment of problems such as the study of potential energy surfaces or direct dynamics.

The **basis set** information are specific of the AO domain, so in a given Q5Cost file several AO definitions are possible. The AO domain (like all the other domains) can be splitted into several sub-domains, each for every different AO choice. The different AO integrals can arise from different basis set or from different orbital types (Atomic Orbitals vs. Symmetry Adapted Linear Combination of Atomic Orbitals)

.

Concerning the **wave function**, complex issues arise because no clear agreement is reached on the many different wave functions that could be stored. A general agreement has been found that the lowest level representation of a wave function is a multi-reference configuration interaction, where each determinant is described together with a proper coefficient. This allows a complete description of the wave function, and should therefore be generally present among the stored information. We do however realize the following points:

1. A multireference description lacks any additional information about the true nature of the wave function (examples like HF, CAS, RAS, CAS+S etc..). Description of higher level information is important to prevent loss. Q5Cost must provide access to proper storing of diversified information to fully qualify the described entity.
2. A design assumption about the concept of Domain is that Properties living into a given Domain have indexes referring to a particular concept belonging to that Domain. As an example, the molecular orbitals one-electron integrals refer to molecular orbitals, and they belong to the MO Domain..

The Library

The new interface for the user provides access for handling geometry and basis set information. Furthermore, storage of the wave function is discussed.

The “Geom” routines allow the client to store and retrieve the molecular geometry. The geometry is considered top level information of the chemical system under investigation, therefore is associated with the System. At present only one System, and consequently only one geometry, per Q5Cost file is allowed. The restriction to a single System per file will be possibly lifted in future releases of the library. This will

require, however, a slightly modified programming interface, and considerations about inter-System characteristics, currently outside the scope of our efforts. The actual data handled by these routines are the Cartesian coordinates, the atomic numbers and the labels for all the atoms, unique by symmetry, contained in the system. The only metadata needed by these routines is the number of atoms. If in future releases other coordinate types are going to be supported in addition to the Cartesian, specific metainformation will be added.

Information about the Basis Set is associated to the AO domain, since multiple basis sets can exist in a given Q5Cost file. Its presence is bound to the presence of a Geometry description. The routines allow the user to store and retrieve the exponents and contraction coefficients of each contracted function defined for each angular momentum on each atom of the system. The actual data handled by the basis set routines are the exponents and the coefficients defining the contraction. The required metadata include the type of angular functions, angular momentum, and the number of functions for each contraction. A possible link to the EMSL data bank [5] is advisable in the future. Basis set names like "6-31G*" could also be an option, however, these names do not represent a standard nor in terms of naming, neither in terms of actual contents among different quantum chemistry programs. As a consequence, the plain use of basis sets' simple name should be discouraged, because it makes difficult to check for the consistency between two stored bases, and hence preventing easy interoperability.

The routines for storing and retrieving the wave function are still a matter of debate in the working group, as better described in the following section. A tentative set of routines are available anyway for a Configuration Interaction type Wave Function (CI-WF), in terms of a linear combination of Slater determinants made of Molecular orbitals. The data to be considered are the coefficients (real numbers) and the Slater determinants that are part of the combination. The metadata needed depends on the type of WF to be described and includes the energy and the core energy associated to the WF, the Molecular orbitals to be used for the determinants and their classification.

The Wave function

The storage of the Wave Function (WF) is a particularly relevant point, and poses different problems because of its size and the wide variety of possible WF's. The wave function is usually a very large object, and its storage often represents the largest amounts of data needs in a quantum-chemistry calculation. Several options are possible, each one typically suited for a particular type of WF calculation.

Usually a WF is called of Configuration Interaction (CI) type if it is expressed as a linear combination of Slater determinants (thereafter "determinants"). Therefore, a CI WF is expressed in terms of a list of Slater determinants and the corresponding coefficients of the linear expansion. We notice that this is, by no means, the only possibility. In fact, many other different WFs are currently used in Quantum Chemistry, depending on the level of theory used to study the system. For instance: Coupled Cluster (CC) Contracted CI (C-CI), Density-Matrix Renormalization Group

(DMRG), and many others. At the moment, we decided to restrict our work on CI WF because the interest of all the partners of the project is on this type of WF. We plan in the future to extend our definition to more general WF's. In particular, CC or C-CI wave functions could be easily stored by simple generalization of the procedure used for a CI WF; some works is in progress on this point. Things are more complex for DMRG, and we do not plan, for the moment, to deal with this kind of formalism.

Even with this restriction, the different ways a WF can be defined are rather numerous.

1. **SCF** (or HF): If a single determinant is used to describe the WF, this is called of Self Consistent Field (SCF), or, equivalently, Hartree-Fock (HF) type. One should notice that the Molecular Orbitals are usually variationally optimized in order to define an SCF WF.
2. **FCI**: If all the Slater determinants are considered, the WF is called of Full Configuration Interaction (FCI) type.
3. **CAS**: In the Complete Active Space Self-Consistent Field (CAS-SCF) formalism, the orbitals are classified into three disjoint classes whose union gives the total set of orbitals:
   - Occupied Orbitals (O), if they are doubly occupied in all CAS determinants;
   - Virtual Orbitals (V), if they are empty in all CAS determinants:
   - Active Orbitals (A), if their occupation varies in the set of the CAS determinants, i.e., each A orbital can be either singly or doubly occupied or empty in the CAS determinants.
4. It is important to note that in the special case when all the orbitals are in the Active class (O and V classes empty) the CAS-type WF degenerate into a FCI WF. In other words, the FCI type WF is a special case of the more general CAS-type WF when the O and V classes are empty. On the other hand, when the A class is empty and all orbitals are either in the Occupied or Virtual classes, the CAS-type WF become an SCF one. In conclusion we can assume the CAS-type WF to be a more general type of CI wave function with FCI and SCF wave functions as special cases.
5. **TCI**: in the Truncated CI (TCI) formalism, the determinants are obtained by exciting a given number of electrons from a set of "Reference" determinants. In the most common case, the reference determinants have a CAS structure, and the excitations are restricted to Singles (S) or Singles+Doubles (SD). One obtains then the CAS-S and CAS-SD CI spaces, respectively.
6. **Selected-CI**: in this case the set of determinants can be completely arbitrary. This is the case, for instance, if the determinants are derived from a TCI or a FCI expansion by selecting those determinants whose coefficients have an absolute value larger than a given threshold.

In the first four cases, it is possible to define an ordering of the Slater determinants, and the WF is completely defined by specifying

– The WF nature (SCF, FCI, CAS-CI, etc), and the corresponding information that uniquely specify it (level of truncation in TCI, nature of the CAS, etc)

– The ordering used for the determinants;

– The corresponding CI coefficients.

In the other case, described at point 5., when this is not possible, for example because the determinants involved in the CI expansion are completely arbitrary or their generation schema is too complex, it is necessary to explicitly define all the determinants. In order to describe the CI WF from a general point of view, the simplest way is to give the determinants list without relying on a possible order. The price to pay, of course, is that this piece of information can be very large and space demanding but also redundant in cases like WF's of type 1-4.

Generally speaking, the WF storage requires two data structures:

1. the list of the determinants used to define the WF;
2. the corresponding list of the coefficients of the CI expansion.

The second structure is simply a vector of real numbers, while the first one is a vector of more complicated objects, since there are several not trivial ways to code a Slater determinant.

The first and common way to define a Slater determinant on the basis of Molecular Orbitals is to indicate, for each electron in the system, the spin orbital that hosts it, as shown in Figure 2.

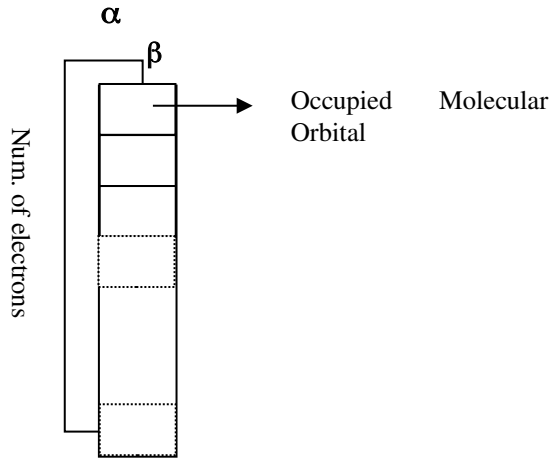

Fig. 2. Data structure for defining a Slater determinant based on the electrons in the system

A similar way, that sometimes can be more effective, is to use vectors running on the spinorbitals and reporting the occupation (1 if the orbital is occupied by an electron, 0 if empty), as shown in Figure 3.

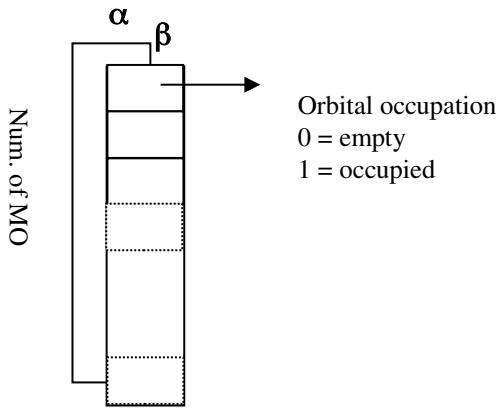

Fig. 3. Data structure for defining a Slater determinant based on the MO in the system

When working with a WF expanded on determinants obtained by an excitation process on CAS determinants, a hole/particle formalism is more effective to describe the occupation of the orbitals in the Occupied and Virtual classes. In this formalism it is advisable to define a reference occupation pattern in the Occupied and Virtual classes (the trivial choice is occupation 2 and 0 for the two classes, respectively), and simply define the occupation in these classes of any other determinant. The differences can be "holes" (an orbital that contained an electron in the reference is now empty) or "particles" (an orbital that was empty in the reference is now full).

Quite often this formalism is used to take care of the orbitals that are in the Occupied or Virtual classes. In the O class, the number of holes n_h in spinorbitals is specified, followed by the list of the corresponding holes:

`n_h, (h(i),i=1, n_h)`

The same is done for the spinorbitals in the V class where the number of particles np is specified, followed by the list of particles:

`n_p, (p(i),i=1,n_p)`

Utilities

A number of utilities have been set up to facilitate the use of the Q5Cost library and data format. The utilities are addressed to the library developers or to the library installers; some of the utilities are for the use of the final users, the programmers that

need to include calls to the routines in their programs and manage the data files produced by a QC calculation.

The user interface to the data format is in Fortran 90, because of a great flexibility of this language and its large use for chemical programming. Nevertheless, other interfaces are available for those programmers which prefer Fortran 77 or other languages. A Python script reads the library code and builds automatically an interface for calling Q5Cost functions in Fortran 77 codes, which exposes the library functionalities to Fortran77 and C programmers. The same script builds also a C++ interface for the use of the library in a C++ environment which is very common in visualization software. In turn, starting from the C++ interface and using SWIG [6], another interface is built for the Python programming language, a well suited environment for writing wrappers.

In the previous releases, we remarked that the users had some difficulties to link all the necessary libraries (HDF5, HDF5-fortran, libz, libm, Q5Cost) in their codes, because the linking process needs a specific order of the libraries in the command line. To simplify the linking of the codes, we chose to put all the needed libraries into one big static library, containing also the Fortran77 and the C++ interfaces. Now, only one file needs to be added in the linking process.

The user documentation is automatically extracted from the source code by a Python tool that can generate by choice a standard HTML file or a text file. This tool is designed in a modular way in order to make it possible to include new formats when needed.

As far as the installation of the library on personal systems is concerned, an automatic configuration script is now available to prepare the library makefile. It takes into account the specificity of the computing platform, like the compilers, the location of the libraries, and verifies if the needed underlying HDF5 layer is present. This configuration script is based on *autoconf* , an extensible package of macros that produces shell scripts to automatically configure software source code packages and guarantees the portability on different architectures; *autoconf* has been developed within the GNU project [7]

Two useful utilities for the final user are available to inspect the q5 file without writing an ad-hoc program. As HDF5 is a binary format, one can argue that the 'small data' such as the molecular geometry, the basis set or the molecular orbitals can't be read easily by humans. On the contrary, since it is based on HDF5, it contains its own description and it is quite easy, using standard HDF5 instruments, to extract a complete description in a human readable form.

- *q5dump* is a simple tool that reports the content of a Q5Cost file, listing the data and the metadata contained in it.
- *q5edit* is a more complete and attractive tool. Using this Python interactive program, the user can navigate through the file, listing the content and modifying the data using his favorite text editor, defined by an environment variable. The Python interface to Q5Cost is used to access the file, and the "*ncurses*" library [8] is used for the interaction of the user with the program.

Wrappers

The Q5Cost library, as a tool for helping data exchange among different chemistry programs, has been used to build specific translation programs [3] to/from the proprietary data format of each of the involved programs. These translation programs, also called "wrappers", are small programs that, using both the program specific I/O library and the Q5Cost library, take care of converting the proprietary data format into the Q5Cost data format and vice versa.

Several QC codes have been wrapped up to now, both general purpose and more specific "home-made" ones usually written and maintained by the partners of the project. In the first class we recall for instance Dalton [9], Molcas [10] and GAMESS US [11,12], in the second class the Bologna FCI [13, 14] code and the Toulouse chain [15, 16] among others. In many cases the wrappers have been written in collaboration with research groups outside the authors of this paper. Here we are simply giving a brief summary, while a more exhaustive and detailed description will be the subject of a forthcoming future paper.

Wrappers are, anyway, normally written and maintained by the author of the chemistry programs. They are external programs that take care to translate the proprietary data to the Q5Cost format (output wrapper) and the common format to the proprietary format (input wrapper). In other cases, the authors liked better to include the Q5Cost routines in the code itself, so to allow the program to read and write "natively" the Q5Cost format. This strategy was used, for example, for the FCI code and for the GAMESS US package: it is more simple and direct of the previous one but it requires a greater involvement of the authors of the codes and it could be difficult in the case of large commercial packages.

The programs that can now use the Q5Cost data format are reported in the following list. In some cases the implementation is still experimental, and has not been released yet in the official versions.

- Dalton [9]
- GAMESS US [11, 12]
- Molcas [10]
- Columbus/ACES II [17, 18]
- Molekel [19]
- Toulouse Chain [15, 16]
- Ferrara 4-indices transformation [20]
- Paris 4-indices transformation [21]
- Bologna FCI code [13, 14]
- QMC=Chem

For the Bologna FCI code [13, 14], the Q5Cost format is now the native data format. Thanks to the integration of FCI with the other codes it was possible to face new scientific problems: the study of the basis set superposition error on the Ne dimer [22], the metal insulator transition [23] and the behavior of small alkali metal clusters [24]. The large systems treated involved a FCI space larger than 1 billion determinants and were good benchmark for the Q5Cost data format and library, which proved to be effective, usable and flexible. The performance outcome, both in terms of disk occupation and access time to I/O, was acceptable too, confirming the previous tests [4].

The Toulouse chain [15, 16] has been only partially integrated. The complete support is however strongly required because it will open the way to the integration of an effective 4-indices transformation, like the one developed in Paris [21]. The performance of the integral transformation is of crucial importance when local approaches are involved. GAMESS US [11, 12] is being integrated in collaboration with the authors, by including the Q5Cost routines directly in the program I/O interface. The Dalton [9] package was originally integrated by using an external wrapper, at the moment we are working on the inclusion of the Q5Cost routines in the package itself.

Another activity in place regards Molekel [19], a free access molecular visualization program being developed by the Swiss National Supercomputing Centre (CSCS). The inclusion of a visualization program is valuable since many of the programs in the set do not have any graphical interface. For including Molekel in the workflow the strategy was to write a converted to the OpenBabel format [25], a format that allows an immediate connection not only to Molekel but with several other environments that are already integrated. The Molcas [10] wrapper is in good shape: at the moment, everything can be translated in Q5Cost format but the wave function expansion which will be the next improvement. A wrapper was also written for the QMC=Chem program, which is a Quantum Monte Carlo program developed at LCPQ, Toulouse.

Conclusion

The design and development of the new 1.0 version of the Q5Cost data format and library has been presented. The format and the library have been designed to facilitate the exchange of information between different Quantum Chemistry codes, thus strongly enhancing interoperability between codes. The ultimate goal of the reported activity is the construction of a grid-based distributed meta-laboratory, where multiple QC codes can be used together in a workflow finalized to solve a chemical problem.

The new features of the version 1.0 of the library have been presented: Geometry, Basis set and Wave Function storage, in addition to new tools in support of developers and end users, such as the binding to programming languages different than Fortran 90. A brief analysis of the wrappers and the codes which have been interfaced up to now has been presented too. The data format and the library appear as valuable tools to make the inter code communication possible and easy; some

preliminary applications proved them to be effective, usable and also efficient, with respect to other Input/Output technologies.

The inclusion of the new features, in particular the Wave Function storage, is expected to allow a more widespread diffusion of Q5Cost, with the possibility to face new original scientific problems.

Glossary

- CAS = Complete Active Space
- RAS = Restricted Active Space
- CAS-SCF = Complete Active Space Self-Consistent Field
- SCF = Self-Consistent Field
- HF = Hartree-Fock
- WF = Wave Function
- CI = Configuration Interaction
- SR = Single Reference
- MR = Multi Reference
- FCI = Full Configuration Interaction
- O = Occupied Orbitals
- V = Virtual Orbitals
- A = Active Orbitals
- I = Inactive Orbitals
- CC = Coupled Cluster
- C-CI = Contracted Configuration Interaction
- DMRG = Density-Matrix Renormalization Group

Acknowledgments

Support from COST in Chemistry action D37 is gratefully acknowledged.

This work has been also financed by the Universities of Bologna and Ferrara and by the Italian Ministry of Research and University MIUR under the project PRIN 2006 "*Molecular Quantum Mechanics: Computational Methods and Analysis of Novel Phenomena*".

The French CNR is gratefully acknowledged too for partial funding.

References

[1] E. Rossi, A. Emerson, S. Evangelisti; *Lect. Notes Comput Sci* **2658**, 316- (2003)

[2] HDF5 a general purpose library and file format for storing scientific data. Also available at htt://hdf.ncsa.uiuc.edu/HDF5/
Specification of XML can be found at the site (http://www.w3.org/XML) ; A.Holmer, XML IE5 – Programmere's Reference, Wrox Press, Chicago, IL, USA, 1999

[3] S. Borini, A. Monari, E. Rossi, A. Tajti, C. Angeli, G. L. Bendazzoli, R. Cimiraglia, A. Emerson, S. Evangelisti, D. Maynau, J. Sanchez-Marin, P. G. Szalay; *J. Chem. Inf. Model.* **47**, 1271-1277 (2007)

[4] C. Angeli, G. L. Bendazzoli, S. Borini, R. Cimiraglia, A. Emerson, S. Evangelisti, D. Maynau, A. Monari, E. Rossi, J. Sanchez-Marin, P. G. Szalay, A. Tajti; *Int. J. Quant. Chem*. 107, 2082-2091 (2007)

[5] Extensible Computational Chemistry Environment Basis Set Database, Version 02/25/04, as developed and distributed by the Molecular Science Computing Facility, Environmental and Molecular Sciences Laboratory which is part of the Pacific Northwest Laboratory, P.O. Box 999, Richland, Washington 99352, USA, and funded by the U.S. Department of Energy. The Pacific Northwest Laboratory is a multi-program laboratory operated by Battelle Memorial Institute for the U.S. Department of Energy under contract DE-AC06-76RLO 1830. Contact Karen Schuchardt for further information. http://www.emsl.pnl.gov/forms/basisform.html

[6] SWIG is an interface compiler that connects programs written in C and C++ with scripting languages such as Perl, Python, Ruby, and Tcl. See http://www.swig.org

[7] Gnu autoconf see: http://www.gnu.org/software/autoconf

[8] Gnu ncurses is a programming library allowing the programmer to write text user interfaces in a terminal-independent manner. See http://www.gnu.org/software/ncurses/

[9] DALTON a molecular electronic structure program, Release 2.0 (2005), See http://www.kjiemi.uio.no/software/dalton/dalton.html

[10] K. Andersson, M. Barysz, A. Bernhardsson, M. R . A. Blomberg, Y. Carissan, D. L. Cooper, M. P. Fülscher, L. Gagliardi, C. de Graaf, B. A. Hess, D. Hagberg, G. Karlström, R. Lindh, P.-Å. Malmqvist, T. Nakajima, P. Neogrády, J. Olsen, J. Raab, B. O. Roos, U. Ryde, B. Schimmelpfennig, M. Schütz, L., Seijo, L. Serrano-Andrés, P. E. M. Siegbahn, J. Stålring, T. Thorsteinsson, V. Veryazov, P.-O. Widmark. Molcas version 6.2 University of Lund Sweden. See http://www.teokem.lu.se/molcas/

[11] Gamess-US, General Atomic and Molecular Electronic Structure System" M. W. Schmidt, K. K. Baldridge, J. A. Boatz, S. T. Elbert, M. S. Gordon, J. H. Jensen, S. Koseki, N. Matsunaga, K. A. Nguyen, S. Su, T. L. Windus, M. Dupuis, J. A. Montgomery; *J. Comput. Chem.* **14**, 1347-1363 (1993).

[12] M.S.Gordon, M.W.Schmidt pp. 1167-1189, in "*Theory and Applications of Computational Chemistry: the first forty years*" C. E. Dykstra, G. Frenking, K. S. Kim, G. E. Scuseria (editors), Elsevier, Amsterdam, 2005.

[13] G. L. Bendazzoli S. Evangelisti; *J. Chem Phys.* **98**, 3141 (1993).

[14] L. Gagliardi, G. L. Bendazzoli, S. Evangelisti; *J. Comput. Chem.* **18**, 1329 (1997).
[15] D. Maynau, S. Evangelisti, N. Guihery, C. J. Calzado, J. P. Malrieau; *J. Chem. Phys.* **116**, 10060 (2002).
[16] C. Angeli, S. Evangelisti, R. Cimiraglia, D. Maynau; *J. Chem. Phys.* **117**, 10525 (2002).
[17] Columbus: H. Lischka, R. Shepard, I. Shavitt, R. M. Pitzer, M. Dallos, Th. Müller, P. G. Szalay, F. B. Brown, R. Ahlrichs, H. J. Böhm, A. Chang, D. C. Comeau, R. Gdanitz, H. Dachsel, C. Ehrhardt, M. Ernzerhof, P. Höchtl, S. Irle, G. Kedziora, T. Kovar, V. Parasuk, M. J. M. Pepper, P. Scharf, H. Schiffer, M. Schindler, M. Schüler, M. Seth, E. A. Stahlberg, J.-G. Zhao, S. Yabushita, Z. Zhang, M. Barbatti, S. Matsika, M. Schuurmann, D. R. Yarkony, S. R. Brozell, E. V. Beck, and J.-P. Blaudeau, COLUMBUS, an ab initio electronic structure program, release 5.9.1 (2006).
[18] ACES II: J.F. Stanton, J. Gauss, J.D. Watts, W.J. Lauderdale, and R.J. Bartlett,Int. *J. Quant. Chem. Symp*. 26, 879 (1992)
[19] MOLEKEL 4.0, P. Flükiger, H.P. Lüthi, S. Portmann, J. Weber, Swiss National Supercomputing Centre CSCS, Manno (Switzerland), 2000. See http://www.cscs.ch/moleke
[20] R. Cimiraglia, C. Angeli private communication
[21] P. Rehinardt, private communication
[22] A. Monari, G. L. Bendazzoli, S. Evangelisti, C. Angeli, N. Ben Amor, S. Borini, D. Maynau, E. Rossi; *J. Chem. Theo. Comp.* **3**, 477-485 (2007).
[23] V. Vetere, A. Monari, G. L. Bendazzoli, S. Evangelisti, B. Paulus; *J. Chem. Phys.* **128**, 024701(1-8) (2008)
[24] A. Monari, J. Pitarch-Ruiz, G. L. Bendazzoli, S. Evangelisti, J. Sanchez-Marin; *J. Chem. Theo. Comp. in press* doi://10.1021/ct7003319
[25] OpenBabel: Rajarshi Guha, Michael T. Howard, Geoffrey R. Hutchison, Peter Murray-Rust, Henry Rzepa, Christoph Steinbeck, Joerg Kurt Wegner, Egon Willighagen. "The Blue Obelisk -- Interoperability in Chemical Informatics." *J. Chem. Inf. Model*. **(2006)** 46(3) 991-998