J Mol Model (2005) 11: 532–541
DOI 10.1007/s00894-005-0278-1

ORIGINAL PAPER

Peter Murray-Rust · Henry S. Rzepa
James J. P. Stewart · Yong Zhang

A global resource for computational chemistry

Received: 22 September 2004 / Accepted: 2 May 2005 / Published online: 21 July 2005
© Springer-Verlag 2005

Abstract A modular distributable system has been built for high-throughput computation of molecular structures and properties. It has been used to process 250,000 compounds from the NCI database and to make the results searchable by structures and properties. The IUPAC/NIST InChI specification and algorithm has been used to index the structures and enforce integrity during computation. A number of novel features of the PM5 Hamiltonian were identified as a result of the high-throughput approach. The system and the data can be redistributed and reused and promote the value of computed data as a primary chemical resource

Keywords Workflow · InChI · NCI · Mopac · XML-CML

Introduction

There are over 30 million published chemical compounds and for many of them a three-dimensional (3D) structure is an essential tool in understanding their properties and behavior. Since only a very small proportion (about 1%) have crystal structures published in the primary literature, it is difficult to build structure–property relationships on statistically significant samples. We now believe that it is possible to calculate molecular geometry sufficiently rapidly and precisely that this should no longer be seen as a problem. However this can only be done by automating the process. This needs a change in our attitude to computational chemistry. For many of the "users" (biologists, synthetic chemists, etc.) quantum mechanical calculations have traditionally been seen as arcane, with their formulae guarded by a priesthood of theoretical chemists. The first full open description of the automation of such a computational chemistry task appeared in 1998 [1–4] in which a 124-processor supercomputer was used to compute structures and associated properties for 52,932 molecules[1]. Although other models for access to appropriate computing resources have subsequently appeared [5, 6] the data obtained from such studies remains unavailable in an easily distributable or extensible form, often for technical and business reasons. In general a scientist wishing access to large numbers of computational results would have to build their own system and repeat the operation.

In this article we show how a redistributable and reusable high-throughput system can be built, and how the results can be searched and reused by simple web-based methods. This opens up the use of computational results as a complement to collections of experimental data.

Electronic Supplementary Material Supplementary material is available for this article at http://dx.doi.org/10.1007/s00894-005-0278-1

P. Murray-Rust · Y. Zhang
Department of Chemistry, Unilever Centre for Molecular Informatics, University of Cambridge, Lensfield Road, Cambridge, CB2 1EW, UK

H. S. Rzepa (✉)
Department of Chemistry, Imperial College, London, SW7 2AY, UK
E-mail: h.rzepa@imperial.ac.uk
Tel.: +44-778-6268220
Fax: +44-20-7594 5804

J. J. P. Stewart
Stewart Computational Chemistry, Colorado springs, CO, USA

Achieving a global extensible computational chemistry resource

The idea of reusable calculations goes back many years, but in practice there are no systems that support it. This is illustrated by the lack of common formats for computational input/output and the lack of a culture for depositing full data at publication. Indeed, even a

[1]Earlier implementations of a commercial server-based computation and database systems were described only in the form of abstracts.

significant proportion of summary data is lost, as shown by a quote from a recent article [7]:

> One hundred and five optimized geometries had been obtained in the present work [...] Reporting all of these optimized geometries is nonsense. [...] Therefore, ... optimized geometries for nine studied isomers are reported in Tables 1, 2 and 3 as an example.

Such selective tabulation of results is not atypical, and it is worth considering what this tells us about the publication process. Most journals (including the one where the above was published) have had supporting or supplemental information schemes for some time. The following extracts from author submission guidelines for such material are representative:

> Supporting Information should be formatted to fit within a minimum number of pages, e.g., text and tabular material should be double spaced and graphics should be reduced to a size that still allows clear viewing over the Web ... The page size should be (U.S. Letter) ... All pages should be numbered consecutively starting with page S1... All Supporting Information files of the same type should be submitted as a single file (rather than submitting a series of files containing individual images or structures). Where appropriate, supporting information should be consolidated into a single word-processing file...

The emphasis on the “printable page” as a unit of information in these instructions often takes precedence over facilitating or encouraging reuse of the data in a computer-readable or semantic sense, and hence can result in the loss of data as illustrated above.

In this study the results are envisaged as being reusable in several ways:

– A chemist could submit new molecules to a web-server which will optimize their structures and compute properties. The results would then be added to the archive in an institutional repository. This is the chemical equivalent of peer-to-peer sharing, where it is possible to offer, e.g., semiempirical quantum mechanical calculations by “payment” through the exchange of complementary information. The concept involves the user submitting a molecule and its 3D coordinates along with, say, its melting point and optical rotation as a “payment” and receiving computational results in return. These data are then integrated with the submitted information and made openly available to the global community. Anyone with molecular information would be encouraged to use this service and thereby to add to the communality of resources. In this spirit all our software (but not third-party) is OpenSource [8, 9][2].

[2]Earlier implementations of a commercial server-based computation and database systems were described only in the form of abstracts.

– The computational methods in the system can be replaced so that different methodologies can be used for existing compounds. The results can be reused independently of the original motivation for calculation, for example,
  – As starting points for further theoretical chemical calculations
  – As building blocks for geometry of complex molecules, and e.g., extraction of pharmacophores
  – Large classes of closely related compounds can be used to give systematics about the effects of substitution on geometry and charge distributions
  – The outliers in the distribution (geometry, charges and properties) can be identified and examined manually. These can give new chemical insights and suggest new systems for systematic study
  – The creation of a library of fragments
  – The systematic examination of a large and consistent collection of chemical computation for trends in convergence, pathological behavior, etc.

– Such computations, carried out by an exposed protocol and with universal metadata, are by themselves reusable objects of value to the community. A collection of such objects should be regarded in the same way as a database of experimental data. The protocol itself leads directly to improvements in the methodology in (at least) QM calculations. Variation from experimental data or internal variance may suggest places where the methodology can be refined.
– Creation of an RSS-based “news feed” [10] as the calculations are archived.
– Web-search for individual archived calculations, including chemical structure, followed if desired by download of appropriate components of the archived calculations.
– The infrastructure itself can be replicated and used at difference sites or for a different sets of molecules. Merging, sorting and reuse of the separate databases is facilitated.

Unlike previous studies, this work was carried out on “free” computers without requiring bespoke software. It used 20 teaching machines during a vacation and a system that can be easily replicated and copied. Relatively few copies of the system would be able to calculate most of the published 10–15 million molecules and would still represent a trivial fraction of the global chemical computational resource [5, 6].

Methods and materials

What follows describes the creation of a resource based on 0.25 million molecules [11] with initial 3D structures optimized using the PM5 semiempirical method [12][3]. A key strategy is that the same methodology (protocol) was applied to each molecule. The results of this are publicly available in XML format [13, 14].

The molecule input set

The collection of compounds as assembled by the US National Cancer Institute (NCI) over a 30 year period of screening was chosen. This dataset is in the public domain and is widely used in molecular modeling [11].[3] One motivation of the current work was to repay this public effort by enhancing it with additional calculated properties. The NCI entries had a wide chemical range, although there was no initial documentation of the type and frequency of different molecular classes. It was decided that at the outset, all molecules would be submitted to the calculation and then the data set would be filtered and the protocol adjusted accordingly.

Each compound has a unique identifier (NCI or NSC number), a 2D connection table (structural diagram) and a (single) set of 3D coordinates. As with the previous study [1] no attempt was made in the present study to address multiple conformations of these molecules and the coordinates represent *a* conformer (not necessarily either a low energy or a global minimum) generated by the Corina program [15]. The 2D and 3D files were distinct and, as discovered in retrospect, do not always correspond precisely. In particular for ionic salts the 3D structure contained only one ion, usually the largest.

The creation of a searchable database has been made possible through the recent development of unique chemical identifiers in the IUPAC/NIST task force (InChI) [16, 17]. This is provided by an algorithm which normalizes the description of the connectivity of a molecule, and creates a canonical representation. Because the InChI approach is independent of formal bond order and depends only on the nature and connectivity of atoms, it is ideally suited to the representation of this type of computational study. Every molecule, therefore, generates a characteristic InChI and this can be used not only to index them but to determine whether two connection tables are different. It is important to note that the InChI is computed only from the molecular structure and is not absolutely bound to names or organizational identifiers. Thus in the current NCI collection CH3C(=NOH)C(=NOH)CH3 is associated with the name "NSC9", and many names such as "Biacetyldioxime" and "Dimethylglyoxime". If later the structure for "NSC9" is found to be wrong (i.e., NSC9 does not correspond to CH3C(=NOH)C(=NOH)CH3), the association is removed. Our indexing depends on the molecular structure and its connection table rather than being based on NSC numbers or names. In this way all of the current work would still be valid even if NSC numbers were revised or withdrawn. Importantly it acts as a globally agreed identifier with a similar function to the recently proposed Life Sciences Identifier (LSID) [18].

Because the NCI data set has been built over many years, the quality varies considerably [19]. This does not affect the current work since the only information used is the 3D coordinates and the unique identifier generated from them. We attach the NCI number but stress that it only corresponds to the assignment current in NCI in 2003 CE. If readers wish to link into samples or screening data, they should consult NCI records. If these caveats are carefully observed, however, readers and reusers should be able to use our calculated properties to add to any studies done with the NCI data set. No further "cleaning" of these NCI molecules was attempted prior to MOPAC calculation. In particular, all molecules computed are closed shell systems. There are no cases where molecules differ solely by charge. There are some cases where a molecule contains cation(s) and anion(s). NCI then (arbitrarily) create 3D coordinates for one of them. We compute this with the number of electrons required by the formula—e.g., "NH_{4+} ", not NH_4. In some cases there may be two entries that describe two species differing by proton count, e.g., NH_3 and NH_4. In this case, we have computed both independently with the appropriate number of electrons. The net charge on the molecule has not been used other than to verify the total electron count.

Methodology

The developmental history is described in some detail, since the architecture was continually redefined whenever understanding was improved or new technology introduced. Some of the initial design caused considerable problems later, and involved more human effort than had been planned. Our final architecture requires much less human effort, but nonetheless this is still more important than the computer resources. The ultimate objective is to develop a system where no human curation is required.

The basic strategy was to find a single protocol that could be applied automatically to all molecules. Although this study used only one code, MOPAC2002, the architecture is designed to be generic and can be used with any computational code (whether based on quantum mechanical or molecular mechanical theories) that accepts coordinates and control parameters. It is based on single units of calculation or jobs (Fig. 1). A job is the minimum unit for which there are logical inputs or outputs (in XML), and could be based on calculations using more than one physical job linked by legacy workflow. In this study an initial decision was taken to define a single physical MOPAC job as comprising 500 independent logical jobs to minimize operational overheads. We emphasize that this leads to new approaches

[3]The NCI database has been previously used for similar purposes, for example to evaluate a proposed bond charge correction model for charge distributions in small molecules.

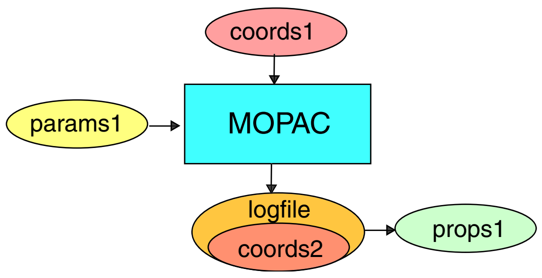

Fig. 1 A single logical MOPAC job

to developing systems where the final approach may be considerably different from the beginning.

Although MOPAC2002 uses a single *physical* ASCII input stream this is abstracted into two *logical* streams serialized as XML:

– The input coordinates (including element types and total charge)
– The control parameters ("the protocol") defining the method, limit of convergence, etc.

In principle the control parameters might depend on the nature of the molecule but throughout this work they were constant for each MOPAC unit. The logical streams are combined through the use of transformations based on XSLT (extensible stylesheet language) [20] to create the flat MOPAC input.

MOPAC2002 results can occur on several streams but in this work only the single logfile, containing the complete history of the calculation, has been used. This output contains the initial configuration, optimization history, and final coordinates with properties (wavefunction, normal modes, charges, etc.) computed at this geometry. From this physical stream, two or more logical streams are extracted as XML.

Our architecture allows jobs to be linked into a *workflow* under the control of a script (such as ant) or workflow engine and an example is shown in logical form (Fig. 2):

Here job1 uses the raw coordinates and the initial protocol params1 to produce an output. If successful this is parsed into XML (see below), combined with params2 and passed into job2 which is similarly chained into job3.

Error handling

It became clear that even 0.01% error rate could not be tolerated so methods had to anticipate all kinds of failure. Any unexpected program exits are detected by the scheduling system used (Condor) [21, 22] the workflow is aborted and the error reported. There were many reasons for such errors.

– The physical job overran time limits
– An abnormal input structure was detected (including physically unrealistic molecular geometries with close approach of two or more atoms)

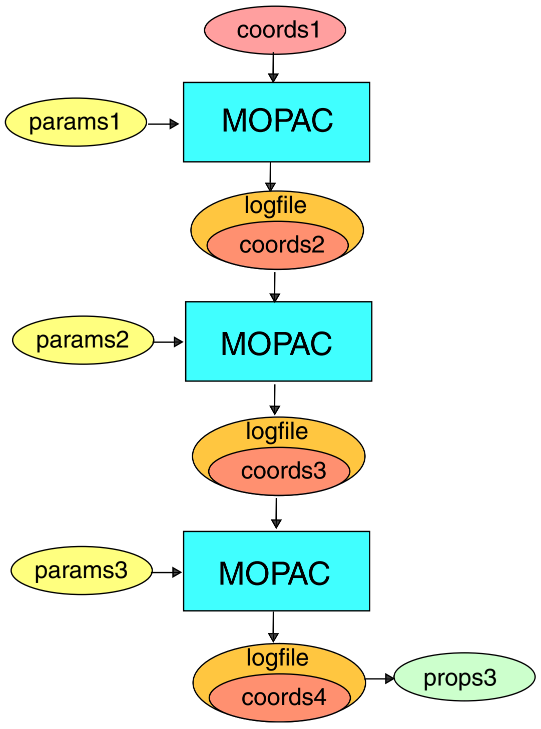

Fig. 2 Workflow for MOPAC jobs

– There were difficulties in convergence or other pathological behavior
– The program crashed. In fact this happened only very rarely (less than five times in 750,000 MOPAC jobs)
– The system crashed. Also very infrequent, causes being machines switched off, worm attack, etc.

Evolution of the protocols

The initial strategy was to use a single protocol to optimize all 250,000 molecules using loose convergence criteria (internal rather than Cartesian coordinates and relatively high final coordinate gradients). This allowed rapid feedback about the nature of the data, the times for computation, the pathology, etc., from which was discovered that;

– About 1% of the molecules were immediately rejected by MOPAC. This was usually due to unusual elements for which PM5 parameters were not available, grossly unreasonable connection tables (e.g., some molecules were decorated with superfluous hydrogen atoms) or errors resulting from unacceptably close contacts between two atoms.
– The times varied from 0.5 to 500,000 s per molecule, a factor of 10^6 . This had an immediate consequence for later jobs. Manual inspection of molecules with large computation times revealed them to be invariably large and many such as oligopeptides were conformational highly flexible. It

is not clear that automatic calculation for such molecules is useful and in later stages they were crudely filtered by runtime limits.

– Certain molecules converged very slowly. In general these again were flexible or had very flat minima. These problems were addressed in later jobs by recalculating the Hessian (second derivative matrix) every 25 or less cycles to provide better search directions for the geometry optimizer.

Initially the jobs were set up manually and to save time 500 physical MOPAC jobs, each containing 500 molecules, were created and a Condor batch system was installed for managing the submission of jobs to PCs as they became free. The batch system was used to avoid latency in starting jobs. The NSC number was not initially contained in the MOPAC job and this caused problems when sub-jobs terminated abnormally. Without the NSC number considerable effort was required to link the output results to the appropriate input. Moreover it was assumed that each batch would run for approximately similar times, but in reality, some batches contained many "expensive" molecules and this disrupted the workflow. Experiments established that the optimum number of molecules submitted per batch job should be reduced to 10–50 to lessen problems with error tracking. The Condor system ran virtually without problems and reduced the human time considerably. With a relatively low time to completion for batches of 10–50 molecules, there was no need to invoke more complex aspects of the operation of Condor such as job checkpointing and restarting.

Late in the project, an Xindice database [23] was incorporated into the protocol. This is a native XML database, i.e., one to which an XML document can be directly added without needing a database schema. Any element in the document can be indexed. In this work documents were indexed through the InChI identifier. The process of obtaining this identifier includes normalization of the structure (e.g., aromaticity, bond order, stereochemistry, etc.), canonicalization and serialization to a unique string. Thus NSC1, methyl benzoquinone (Fig. 3) has a basic InChI of C7H6O2,1H3-5-4H-6(8)2H-3H-7(5)9. InChI can also be used to add canonicalization for charge, tautomers, stereochemistry and isotopes.

The complete XML output of the final jobs was loaded into Xindice [21, 22] and made searchable via the InChI identifier. It is stressed that this identifier is a precise uniquifier for the molecule regardless of its likely stability or reasonableness.

During loading into Xindice, it was discovered that there was much molecule duplication within the database and this was later confirmed in discussions with NCI. It arises from historical problems in checking manual data entry, but most importantly from the way that the 3D structures were generated. Many NCI compounds had two or more molecules as in salts and complexes. The 3D structures retained only the largest entity such as an organic cation or host. Moreover many of the 2D INChIs were different from the 3D INChIs, suggesting problems in the generation of structures. In many cases this is due to ambiguity in the number of hydrogen atoms. Hydrogen rectification/normalization is possible algorithmically, although a general solution is non trivial, and we chose to discard entries with ambiguous hydrogen counts at this stage (whilst recognizing this as a future improvement to the methodology).

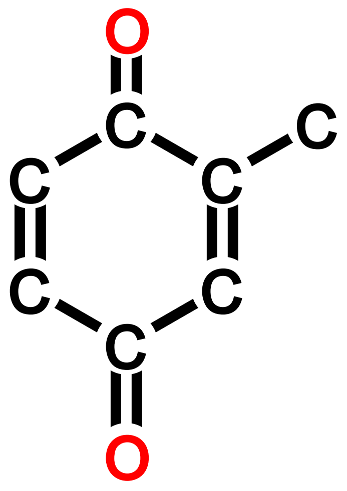

Fig. 3 Carbon framework for methyl benzoquinone

Because of the very large number of structures and their chemical variety, a number of rare problems were encountered. This effectively stress-tested MOPAC and several minor bugs and features in this code were addressed as part of the workflow. We believe such an approach to architecture and dataset design could be a useful tool for testing other codes.

During the process of refining the protocol, it became clear that the first phase had an unacceptably high convergence limit for geometry optimization, so stricter values were only progressively introduced. Fortunately the Condor system was able to "scavenge" more computing cycles than originally envisaged, and the normal levels of precision for single-molecule optimization were ultimately adopted for all the calculations.

An important component of the initial protocol was to include mandatory calculation of the Hessian matrix (the matrix of second derivatives of energy with respect to atom coordinates). Many years of experience have suggested that accidental (or imposed) initial coordinate symmetry may result in unwarranted conclusions regarding the final molecular geometry or symmetry. After the final phase it was noticed that a significant proportion of molecules had one or more negative eigenvalues computed for this Hessian matrix, indicating that the molecule was not at an energy minimum but on a higher order stationary point such as a transition state. A solution to this is to use the "eigenvector following" (EF) algorithm to follow the vectors corresponding to the negative eigenvalues. This should result (in theory) in coordinates corresponding to lower molecular energies devoid of any negative Hessian eigenvalues. Typically,

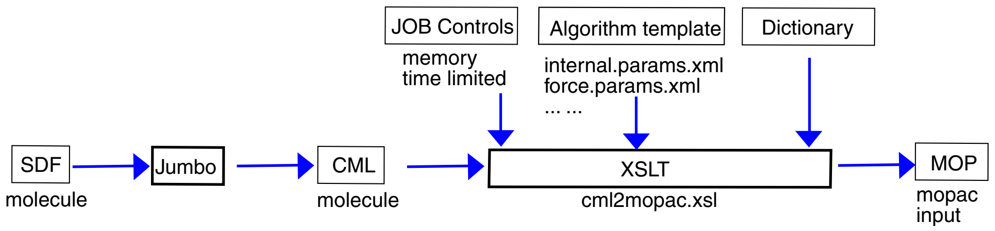

Fig. 4 Workflow schematic for conversion of an SDF file to a Mopac input file

such modes arise from eclipsed methyl or alkyl groups, or from planar starting geometries for intrinsically non-planar groups such as aromatic amines, amides, etc. In practice, although eigenvector following removes a certain proportion of these negative roots, some remain in the final output (due possibly to round-off error in derivative calculation, possibly even a small degree of rotational variance of the wavefunction itself, or unidentified errors in the EF algorithm). XML makes it easy to identify them so that they could be passed to other codes for further analysis.

Physical job input

Data defining a calculation are supplied to MOPAC as a text input file. The first line consists of the following six keywords, four of which have assigned (non-default) values and which control the calculation: T = 3600 RECALC = 8 PRECISE GNORM = 0.1 PM5 charge = 0. The remaining lines correspond to standard MOPAC input. Because the NCI molecules are held in the SDF format, a conversion process is necessary. We use a two stage process in which the SDF files are first converted into CML files using the JUMBO toolkit. A stylesheet (cml2mopac.xsl) is used in the second stage to convert the molecule from CML into the native MOPAC input (*.mop file) by combining with Job Controls and Algorithm template and Dictionary. This structure allows different types of job to be defined using the appropriate Job Controls and Algorithm template (Fig. 4). The job controls used in our calculation are time limitation, specifying how often the Hessian second derivative matrix is recalculated and a normal coordinate analysis following completion of geometry optimization. These are specified as CMLComp [24]:

```
<parameterList role="ccml:control">
 <parameter>
  <scalar dictRef="mopac:timesec">3600</scalar>
 </parameter>
 <parameter>
  <scalar dictRef="mopac:recalc">true</scalar>
 </parameter>
 <parameter>
  <scalar dictRef="mopac:force">true</scalar>
 </parameter>
</parameterList>
```

Physical job output and CMLcomp

Part of a Mopac output stream is shown below;

```
MOPAC2002 (c) Fujitsu
PM5 CALCULATION RESULTS

 MOPAC2002 Version 1.01 CALC.'D. Sun Jun 22 23:23:34 2003
PM5-THE PM5 HAMILTONIAN TO BE USED
  CHARGE ON SYSTEM  =0
PRECISE-CRITERIA TO BE INCREASED BY 100 TIMES
T=-A TIME OF 3600.0 SECONDS REQUESTED
DUMP=N-RESTART FILE WRITTEN EVERY 7200.000 SECONDS
RECALC=-DO 8 CYCLES BETWEEN HESSIAN RECALC
GNORM=-EXIT WHEN GRADIENT NORM DROPS BELOW .100
T=3600 RECALC=8 PRECISE GNORM=0.1 PM5 CHARGE=0
[id=NSC1][pos=1][name=mol1]Use XYZ by default

ATOM NUMBER      CHEMICAL SYMBOL      X (ANGSTROMS)      Y (ANGSTROMS)      Z (ANGSTROMS)

1                O                    0.000290244*       -0.000903121*      -0.015300924*
2                O                    5.347499281*       0.000311673*       -0.007858664*
3                C                    1.213650143*       0.012635295*       0.006474477*
......
HEAT OF FORMATION  =  -44.338287 KCALS/MOLE
ZERO POINT ENERGY 69.798 KCAL/MOL
......
COMPUTATION TIME  =11.33 SECONDS
WALL CLOCK TIME  =13 SECONDS
......
```

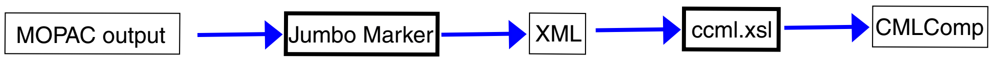

Fig. 5 Workflow schematic for conversion of a Mopac output file to

The structured data contained in this output can be extracted systematically using the regular-expression-based JUMBOMarker [24]. We note that these regular expressions are easily modified to respond to version changes in the program output (Fig. 5). Such changes are notorious for breaking hard-coded output parsers (post-processors).

```
<module start="-1" end="-1" id="rootModule">
 <module start="0" end="1911527">
  <module start="0" end="287" name="jobList" id="job-
List" splitBefore="header">
   <module start="0" end="276" name="job" id="job">
    <module start="0" end="9" ref="header" id="hea-
der.ref">
     <scalar dictRef="ccml:program">MOPAC2002 (c) Fu-
jitsu</scalar>
     <scalar dictRef="ccml:method">PM5</scalar>
     <scalar dictRef="ccml:program">MOPAC2002</scalar>
......
    <module start="196" end="218" ref="cartesians2" id="-
cartesians2.ref.2">
     <molecule id="NSC1" name="mol1">
      <atomArray name="cart2.aa" ref="cartesians2" id="car-
tesians2.ref.2">
      <atom id="a0" elementType="O" x3="0.010011367"
y3="-0.106728156" z3="0.025210937"></atom>
      <atom id="a1" elementType="O" x3="5.352140349"
y3="0.085092937" z3="-0.027419739"></atom>
......
</module>
```

The molecular properties computed by MOPAC (or any other modeling package) can be captured into the CMLComp [24] file by use of appropriate dictionary references:

```
<?xml version="1.0" encoding="UTF-8"?>
   <job id="NSC1">
 <module role="init">
  <module role="header">
   <list>
    <scalar dictRef="ccml:program">MOPAC2002
       (c) Fujitsu</scalar>
    <scalar dictRef="ccml:method">PM5</scalar>
    <scalar dictRef="ccml:program">MOPAC2002</scalar>
    <scalar dictRef="ccml:version">1.01</scalar>
    <scalar dictRef="ccml:wday">Sun</scalar>
    <scalar dictRef="ccml:month">Jun</scalar>
    <scalar dictRef="ccml:day">22</scalar>
    <scalar>23</scalar>
    <scalar dictRef="ccml:min">23</scalar>
    <scalar dictRef="ccml:sec">34</scalar>
    <scalar dictRef="ccml:year">2003</scalar>
 <list>
</module>
......
<molecule id="NSC1" formalCharge="0" name="mol1">
  <atomArray name="cart2.aa" ref="cartesians2"
       id="cartesians2.ref.2">
```

```
  <atom id="a0" elementType="O" x3="0.010011367" y3="-
0.106728156" z3="0.025210937"/>
  <atom id="a1" elementType="O" x3="5.352140349"
     y3="0.085092937" z3="-0.027419739"/>
 </atomArray>
 </molecule>
</module>
```

Computational chemistry has many thousands of concepts and it is ultimately important that all of these are captured in XML markup. MOPAC uses a subset of these concepts and for markup only those involved in program input or output are worth retaining, and not those reporting internal states of the program or workflow. The most formal method is to find all FORTRAN READ or WRITE statements and formalize the concepts. This was achieved for MOPAC7, the last available Open Source version, which has about 1,200 such statements. Each concept is supported either by a specific XML element in CML or by a link to an entry in a dictionary. A typical marked up concept is:

```
 <scalar
dictRef="mopac:deltahf" type="xsd:float"
units="unit:kcalmol">23.4</scalar>
```

which tags the value as having a particular type (W3C float), units as entry kcalmol in the unit dictionary and with the semantics defined in entry deltahf in the Mopac dictionary.

Dictionaries are themselves namespaced and can be used to validate the input or output file or interpret the information in the parsed results. A dictionary is a container for entry elements, but can also contain unit-related information. The dictRef attribute on a dictionary element sets a namespace-like prefix, allowing the dictionary to be referenced from within the document. In general dictionaries are referenced from an element using the dictRef attribute, as above. An example of CMLComp dictionary is shown below:

```
<dictionary
 dictRef="ccml"
 title="CCML dictionary"
 xmlns="http://www.xml-cml.org/schema/stmml"
>
......
  <entry id="input" term="Input">
   <annotation>
    <appinfo>
    </appinfo>
   </annotation>
  < Definition >
  The input module
  < Definition >
  < description >
  This may echo some or all of the input parameters and data
  < /description >
 < /entry >
 < entry id = “job” term = “Job” >
  < annotation >
   < appinfo >
   < /appinfo >
  < /annotation >
  < Definition >
  A computational job
  < Definition >
  < description >
  < /description >
 < /entry >
 < entry id = “parameters” term = “Parameters” >
  < annotation >
   < appinfo >
   < /appinfo >
  < /annotation >
  < Definition >
  A group of parameters
  Definition
  < description >
  The role of the parameters is program and context dependent
  < /description >
 < /entry >
......
< /dictionary >
```

Xindice and XPath

Following the above procedure, we generated the InChI identifier for each molecule, inserted this identifier into the CML file and deposited this into a Xindice database [23]. The database was indexed on the InChI elements and molecule ids (as the NSC number). This procedure results in a query time of about 20–60 ms. A typical XPath query syntax has the form:

```
“//entry[//basic[. = ’C7H14S,1H2-3H-5H2-8-7H2-6H2-4H2-2H3’]]”
```

which produces a result from the Xindice repository of the form:

```
< ?xml version = “1.0”? >
< !– There are 1 results! – >
< !– The execution time is 0.051 seconds! – >
< entry xmlns:src = “http://xml.apache.org/xindice/
Query”src:col = “/db/wwmm/nci/entry”
src:key = “nci_001111.xml” >
  < identifier version = “0.932Beta” tautomeric = “0” >
   < basic > C7H14S,1H2-3H-5H2-8-7H2-6H2-4H2-2H3 < /basic >
   < charge/ >
  < /identifier >
  < molecule id = “NSC1111” formalCharge = “0” >
    < formula > C 7 H 14 S 1 < /formula >
    < metadata >
......
```

```
        < module role = “final” >
         < module           xmlns:cml = “http://www.xmlcml.org/dtd/
cml1_0_1” role = “summary” >
          < module  start = “141”  end = “146”  name = “compdetails”
id = “compdetails” >
           < scalar dictRef = “ccml:method” > PM5 < /scalar >
           < scalar      dictRef = “ccml:program” > MOPAC2002 < /sca-
lar >
......
         < /module >
           < scalar dictRef = “ccml:scfcount” > 160 < /scalar >
           < scalar dictRef = “ccml:cputime” units = “units:sec” > 6.09
 < /scalar >
           < scalar dictRef = “ccml:elapsedtime” units = “units:sec” > 13
 < /scalar >
          < list >
         < /module >
        < /module >
       < /metadata >
       < atomArray >
        < atom       xmlns:cml = “http://www.xmlcml.org/dtd/cml1_0_1”
id = “a0” elementType = “C” x3 = “0.051088033”
y3 = “-0.130281998” z3 = “0.020777613” / >
        < atom       xmlns:cml = “http://www.xmlcml.org/dtd/cml1_0_1”
id = “a1” elementType = “C” x3 = “1.358661611”
y3 = “0.015806631” z3 = “-0.052561890” / >
......
       < /atomArray >
       < propertyList >
        < property            xmlns:cml = “http://www.xmlcml.org/dtd/
cml1_0_1” >
         < scalar       dictRef = “ccml:hof”       units = “units:kcal” > -
5.07617 < /scalar >
        < /property >
......
       < /propertyList >
     < /molecule >
    < propertyList/ >
 < /entry >
 < /list >
```

This output could, if necessary be post-processed by further XSLT stylesheets to produce more human-readable outputs in the form of, e.g., XHTML pages.

Overall (web-based) architecture and workflow

With the basic data structures in place, we have been able to set up a “Computing On Demand” black box to collect and compute molecules. The workflow therefore comprises acquiring data (in a legacy format if necessary), submitting the MOPAC calculation and having the calculated result and appropriate metadata deposited into an Xindice database, where it is published to the public (Fig. 6).

Results and discussion

Analysis of the MOPAC calculations

The very large number and variety of the molecules puts an unusual stress on any program used and is thus well

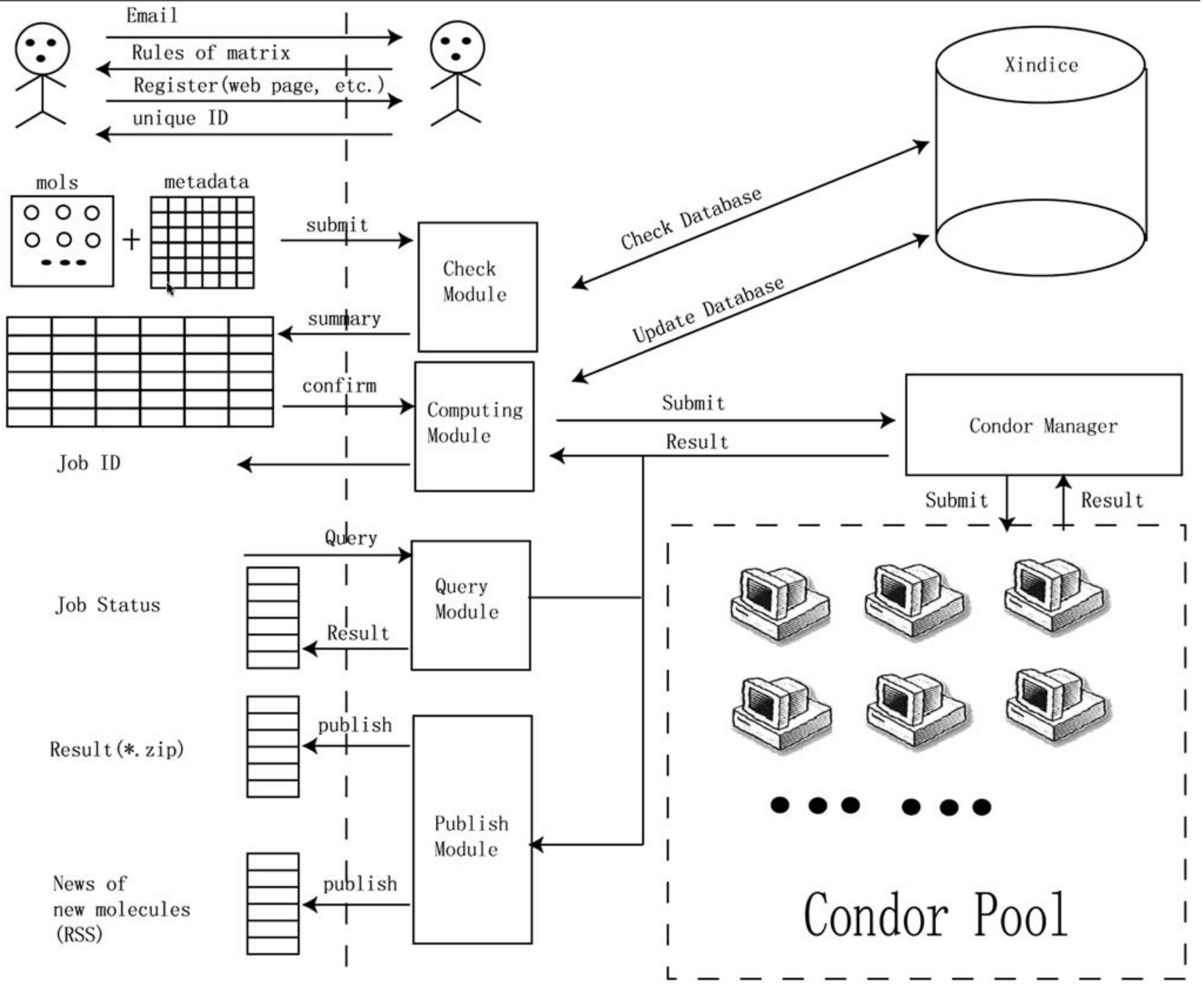

Fig. 6 Overall workflow schematic for the World-Wide Molecular matrix

suited to the detection of systematic features and rare problems. For example one error occurred only once per 64,000 calculations, and the process discovered more "bugs" (albeit minor) than had been reported by the general user community.

The high throughput of molecular geometry optimization sustainable for the NCI database allows some general conclusions to be drawn for the benefits and value of this procedure. For each molecule, a direct comparison can be made of the InChI identifier computed before and after optimization. This process highlights differences in the molecular connection table resulting from the MOPAC optimization, something that is inherently possible in a quantum mechanical approach in which bonds are defined from electron densities rather than arbitrarily defined connectivity. We note for example that this differs from a geometry optimization based on molecular mechanics force fields, from which the initially defined connectivity will always be maintained. Molecules for which the InChI identifier was detected to differ clustered into the following types (see supporting information for examples).

1. Molecules containing a metal atom (Pb, Sn, Ge, Bi, Sb, As, Te and Se), where an increase in coordination is observed. These may be either representational differences or real errors in the MOPAC calculations.
2. Molecules that split into separate fragments (for which an InChI identifier is separately computed) resulting from the cleavage of one or more bonds.
3. Molecules for which the starting structure was wrong/impossible. They include highly bonded atoms, missing charges or missing hydrogens from the original specification. It is worth noting that MOPAC applies no initial checksum for molecule integrity, such as consistency between declared spin state (singlet, doublet and triplet) and the inferred electron count. Providing information about the spin

state would probably allow detection of a significant proportion of such errors, but since this information is rarely declared explicitly it would also require heuristics to allow its automated inference.

4. Molecules where the input specification was clearly in error. Whilst it is impossible to know what the original authors meant, MOPAC has produced an entirely reasonable final geometry. Examples are interpenetrating groups, C–O–O rings instead of esters, etc.
5. Molecules for which the input is reasonable and so is the result but they are different. Examples include tautomers, internal H-bonding connectivity and changes in coordination of the central atom, particularly of electron deficient atoms such as boron, where the conventional structural representation may in fact differ significantly from the more correct quantum mechanical solution.
6. Molecules for which the input is reasonable, but for which the output is not. These examples cluster around elements such as S (and S = O), Iodine and Cl = O systems, and may be clearly revealing significant errors in the MOPAC PM5 parameterizations [12] (or basis set) for these specific elements.

Conclusions

We have described a set of infrastructures based on standard protocols and predominantly Open Source software components which can be used to create a reusable global resource for computational chemistry. Taken with concurrent developments such as grid-based resources [25, 26] this represents a further step in the evolution towards a semantic Web [27] in which automated high-throughput processing of chemical data and information in the form of a World-Wide Molecular matrix [8, 9] is made available.

References

1. Beck B, Horn A, Carpenter JE, Clark T (1998) J Chem Inf Comp Sci 38:1214–1217
2. Beck B (2000) Abstracts of papers, 220th ACS National Meeting, Washington DC, USA, August 20–24, COMP-151
3. Pear M, Berstein J, Li C, McDonald R (1996) Book of abstracts, 212th ACS National Meeting, Orlando, FL, August 25–29, CINF-028
4. Brown RD, Guner OF, Hahn M, Li H (1998) Book of abstracts, 216th ACS National Meeting, Boston, August 23–27, CINF-051
5. Richards WG (2002) Innovation: nature reviews drug discovery 1:551–555
6. Hawick KA, Grove DA, Coddington PD, Buntine MA (2000) Int J Chem article 4
7. Hela M, Yousef YA, Afeneh AT (2002) J Comp Chem 23:966–976
8. Zhang Y, Glen RC, Murray-Rust P, Rzepa HS, Townsend JA (2004) Abstracts of papers, 227th ACS National Meeting, Anaheim, CA, USA, March 28–April 1, CINF-032
9. Murray-Rust P, Rzepa HS, Zhang Y (2004) Org Biomol Chem 2:3192–3203
10. Murray-Rust P, Rzepa HS, Williamson MJ, Willighagen EL (2004) J Chem Inf Comp Sci 44:462–469
11. Jakalian A, Jack D, Bayly CI (2000) Abstracts of papers. Am Chem Soc COMP-001
12. Stewart JJP (2004) Abstracts of papers. Am Chem Soc, COMP-004
13. Murray-Rust P, Rzepa HS (1999) J Chem Inf Comp Sci 39:928–942
14. Murray-Rust P, Rzepa HS (2003) J Chem Inf Comp Sci 43:757–772
15. Sadowski J, Gasteiger J (1993) Chem Rev 93:2567–2581
16. Linstrom PJ, Tchekhovskoi DV (2003) Abstracts of papers. Am Chem Soc, CINF-085
17. Stein SE, Heller SR, Tchekhovski D (2003) Nimes International Chemical Information Conference Proceedings, pp 131–143
18. Clark T, Martin S, Liefeld T (2004) Brief Bioinform 5:59–70
19. Godden JW, Stahura FL, Bajorath J (2000) J Chem Inf Comp Sci 40:796–800
20. See http://www.w3c.org/Style/XSL/
21. Tannenbaum T, Wright D, Miller K, Livny M (2002) Condor—a distributed job scheduler. In: Sterling T (ed) Beowulf cluster computing with linux. MIT Press, Cambridge, MA
22. Thain D, Tannenbaum T, and Livny M (2003) "Condor and the grid". In: Fran B, Anthony JGH, Geoffrey F (eds) Grid computing: making the global infrastructure a reality. Wiley, New York
23. See http://xml.apache.org/xindice/
24. For details of CMLComp and other evolving schemas, see http://cml.sourceforge.net/
25. Frey JG (2004) Abstracts of papers, 227th ACS National Meeting, CINF-025
26. Hughes G, Mills H, De Roure D, Frey JG, Moreau L, Schraefel MC, Smith G, Zaluska E (2004) Org Biomol Chem 2:3284–3293
27. Berners-Lee T, Hendler J (2001) Nature 410:1023–1024