



Construction of a robust, large-scale, collaborative database for raw data in computational chemistry: The Collaborative Chemistry Database Tool (CCDBT)

Mingyang Chen, Amanda C. Stott, Shenggang Li, David A. Dixon*

Department of Chemistry, The University of Alabama, Shelby Hall, Box 870336, Tuscaloosa, AL 35487-0336, USA

ARTICLE INFO

Article history:

Received 9 September 2011

Received in revised form

11 December 2011

Accepted 17 December 2011

Available online 28 December 2011

Keywords:

Computational chemistry

Database

Data mining

Massive data

Collaborative

ABSTRACT

A robust metadata database called the Collaborative Chemistry Database Tool (CCDBT) for massive amounts of computational chemistry raw data has been designed and implemented. It performs data synchronization and simultaneously extracts the metadata. Computational chemistry data in various formats from different computing sources, software packages, and users can be parsed into uniform metadata for storage in a MySQL database. Parsing is performed by a parsing pyramid, including parsers written for different levels of data types and sets created by the parser loader after loading parser engines and configurations.

© 2011 Elsevier Inc. All rights reserved.

1. Introduction

As the computing power of current generation supercomputers exceeds the petaflop (floating point operations per second) scale together with the availability of computational chemistry programs running efficiently on such architectures with user-friendly interfaces, computational chemistry calculations can today be performed routinely with data generated at a significantly higher rate than in the past. Even for a single user, the data are often generated and stored on multiple physically separated systems, both local and remote, with different architectures. Thus, the management of the data without additional software assistance can be very time-consuming, and an improved long-term storage strategy incorporating data analysis must be developed.

Raw output files from electronic structure simulations generated and stored by a single user are often not managed in consistent fashion for easy data retrieval, making it very difficult and costly to be searched and re-used by the same user as well as other users if this has to be done manually; this significantly diminishes the practical value of the computational data over time. Data can also become corrupted or unreadable due to system or software upgrades. Furthermore, users usually adopt intuitive names (to the

user) rather than a uniform naming convention for their data (e.g., chemical compounds), making the organization of the data a daunting task especially when computational outputs are so varied. It is thus quite difficult for a user to search even for his/her own data. This becomes significantly more difficult when other users in a research group want to access data generated by a user who has left the group, for example. As examples, consider the naming of chemical compounds that have not yet synthesized, those with multiple structures and/or electronic states, and complex molecular systems such as those involved in heterogeneous catalytic reactions. How does one properly name a model system and convey what the structure means in a name? Although there has been substantial research on the development of computational infrastructures for chemical identification such as IUPAC's International Chemical Identifier (InCHI) [1], the average user does not usually employ a correct naming convention for their output files. Considering the massive amount of data to be managed, it is crucial to develop a software tool to generate and store well-ordered metadata out of the less ordered raw data, which can liberate users from tedious bookkeeping, and allow them to focus on solving the actual chemical problem of interest. To address this issue, data mining and database techniques must be combined. For such software to be successful, one must not only consider the appropriate computer science techniques in solving the problem, but also consider the practical needs of the users, in this case the computational chemist. The issue of using data generated by others is further exacerbated for the non-expert experimental chemist who wishes to use computational electronic structure data to analyze his/her experiments.

* Corresponding author at: Department of Chemistry, The University of Alabama, 250 Hackberry Lane, Shelby Hall, Box 870336, Tuscaloosa, AL 35487-0336, USA. Tel.: +1 205 348 8441; fax: +1 205 348 4704.

E-mail address: dadixon@bama.ua.edu (D.A. Dixon).

A number of approaches have been developed to address the above problem for computational chemistry, including commercial data management systems such as the SEURAT [2] suite and programs developed by academic researchers such as ChemDataBase [3]. An example of a freely distributed code is Ecce, the Extensible Computational Chemistry Environment developed in the William E. Wiley Environmental Molecular Sciences Laboratory at the Pacific Northwest National Laboratory [4]. Ecce includes an Organizer component for data management but the current version is focused on single job input, running, and analysis [5]. An example of a data management/analysis tool for kinetics is the PRiME (Process Informatics Model) project for developing predictive models of chemical reaction systems PRiME collects and stores data in a data Depository, a library for storing evaluated data, and a set of tools to process the data and use it kinetics models [6]. The PRiME user is responsible for uploading the data to be evaluated and must be registered by a manager of the organization. Some of these packages are of high quality, with friendly graphical user interfaces (GUI) and multi-functional visualization tools. However, the practical needs of the computational chemistry users are often not well-considered. For example, average users would like to have their data automatically managed for them once the data has been generated. Many of the existing computational chemistry database solutions require the users to manually build the database. This is not a problem if a user has a very limited amount of data, in which case there is little practical need for such a database program. However, as discussed earlier, a user often generates a large number of raw output files in a short period of time, leading to a substantial accumulation of data over time, easily thousands to tens of thousands of computational output files annually. For a reasonably sized research group, the amount of raw output files generated annually can be massive. Usually it is necessary for one user to access the data generated by other users in order to not recalculate the same information and to obtain information on computational cost. In addition, a group can include visiting scientists and faculty, post-doctoral fellows, and graduate and undergraduate students. The various researchers can have very different tenures in a group as they leave upon graduation or move on to different projects and it is often necessary for current group members to retrieve data from users who are no longer present in the group. Furthermore, it is highly desirable for the users to not worry about the technical details of the database but rather to focus on the actual chemical problems to be solved. Hence, a smart “housekeeper”, which can deal with all of the technical details of the database construction, is needed.

For computational chemistry, a robust database solution for data storage and management has the following requirements: (a) the database needs to be automatically built by the computer system due to the massive amount of data to be handled; (b) the database needs to be expandable and able to “grow” by itself as new raw data are generated by the users and retrieved and processed by the database program; (c) an extensible implementation is required for new output formats be readily supported; and (d) open source software should be used to develop the database solution so that it can be widely adopted and easily customized for an individual group's needs. In addition, it is not enough for the software to provide just a simple, chemist-friendly GUI with appropriate visualization tools; the software must be able to manage and present all of the data including the metadata. We define robust as the ability of the software to require minimal user intervention, to run on a range of platforms, and be easy to manage.

In this work, we designed and implemented a database solution specifically for computational chemistry, the Collaborative Chemistry Database Tool (CCDBT). Our database solution involves the following tasks or components: (a) a multi-layer “group-user-account” access management system; (b) data synchronization

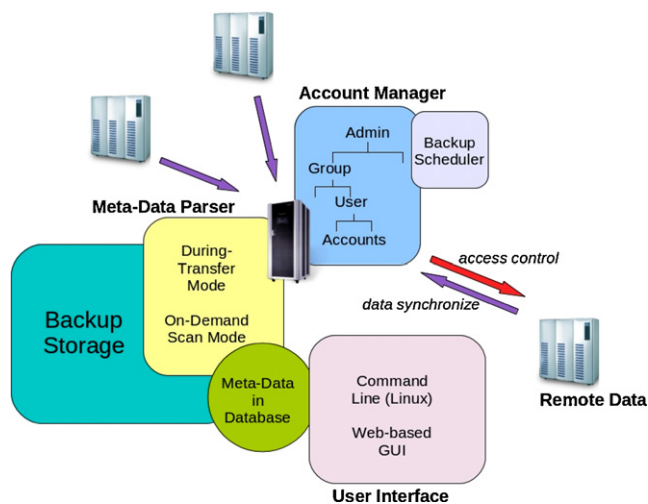


Fig. 1. The CCDBT architecture.

with multiple remote computing hosts; (c) data storage in a multi-layer “group-user-account” hierarchy for multiple users (each of whom could own multiple accounts associated with different remote computing clusters); (d) metadata extraction and storage; (e) metadata query; (f) metadata display; and (g) metadata interpreting and manipulation.

2. Database design and implementation

2.1. General design principles

Python [7] was selected as the implementation language for the CCDBT database solution due to its powerful features. Python is a platform-independent interpreted language. Users of various platforms can directly modify the CCDBT source code to change or improve its features without worrying about compilation and recompilation. Python is also object-oriented, so that new features can be easily implemented. It uses a dynamic data typing method called “duck typing”, which provides the flexibility and accessibility in handling the metadata. The open source database software MySQL [8] was chosen to store the extracted metadata, which is powerful, reliable, and has a well defined application programming interface (API) for Python.

CCDBT was written from scratch using the Python standard language features. Third and -party open source Python modules were used only when there was neither an existing module in the Python standard library nor a facile implementation. The third-party Python modules used in our code include Paramiko [9], MySQLdb [10], and the Python Cryptography Toolkit (Crypto) [11]. Paramiko is a Python module that defines interfaces for the secure shell protocol version 2 (SSH2) [12], and is used for file transfer via the secure file transfer protocol (SFTP) [13]. MySQLdb is the Python API for MySQL. Crypto is a collection of algorithms and protocols for cryptographic uses in Python. The web-based user interface is developed using Python with the Common Gateway Interface support (cgi) module in the standard library, with the integration of the JMOL java applet [14] for molecular visualization.

The CCDBT database application, whose architecture is shown in Fig. 1, checks and synchronizes with the registered remote servers for new files, and then extracts, stores, and feeds the metadata to the appropriate terminals as the stream in the pipeline. The database system is composed of three major components: (a) the profile (account) manager and the file synchronization utility; (b) the metadata generation utility; and (c) the user interface.

2.2. Account management and file synchronization utility

The account manager is designed to meet the needs of cross-group collaborations. It stores a hierarchy of the ownerships and privileges of the user accounts for data sharing and proprietary use purposes. All members in the account manager are password protected, using the encryption feature of Crypto. Administrator, group, user, and account are the main elements of the account manager. The administrator is the run-time manager of the application with the highest privilege including adding/deleting users/groups, resetting user passwords, changing the global setting of the system, and issuing commands for the system as a whole. A user name is linked to a researcher who owns computing accounts on remote supercomputing clusters, and has the privilege to add, delete, and maintain its accounts in CCDBT, which are associated with the accounts on remote servers. The login password for file synchronization for each CCDBT user account (equivalent to the password for the actual account on the remote server) is encrypted before being stored, and can be accessed only by that particular user. “Group” is an attribute of the user allowing users of the same group to share common features and privileges. Moreover, the users and groups in the CCDBT account manager are linked with the users and groups as defined by the server running the CCDBT application, so that each data file and its correspondent metafile have the same ownership and accessibility.

The data files are synchronized from the remote servers to the storage facility, which can be performed on a periodic basis using a task scheduler such as CRON in LINUX, or on demand by the user. Files are stored in the same tree-like hierarchy structure as the account profiles are kept in the account manager, i.e., files are mirrored into nested paths in the storage, e.g., “root/group/user/account/directory&file”. Files are only transferred once as long as they remain unmodified on the remote server. This minimizes the time for file transfer and reduces redundant metadata entries. To achieve this behavior, we applied an algorithm whereby the program recursively walks through both the remote directory containing the data to be transferred and the local path where the data needs to be stored, matches both local and remote files under the same relative file path, and only transfers new or modified files. Descriptors defined by string patterns (e.g., wildcards) can be applied in order to perform synchronization for selected files. Also, the hierarchy in the account manager ensures that different commands can be issued from different levels of the profile tree, thus providing the system administrator with enough flexibility to perform the necessary tasks when needed.

2.3. Metadata parsing

There are many available computational chemistry software program suites, and each might have several versions and builds. Thus, there are various data formats to be considered even if one is only working with the most popular program packages. It is more important to develop a user-friendly and universal parser builder tool rather than to write several parsers for individual versions of computational chemistry software codes (Fig. 2). Our solution is to predefine and implement a parsing engine library which includes basic parsing functions and utilities for data handling and extraction, for example, a function to find a text line that matches the case-ignored string pattern A in the paragraph between the appearance of the string patterns B and C. A specific parser is then assembled as the parser builder chooses particular parsing engines that load the user-defined parameters from the configuration files. The configuration files are in a simple plain-text format, in which one only needs to specify parameters taken by the parsing engines such as string patterns and Boolean true/false. Most of the properties in the raw data file can be extracted into

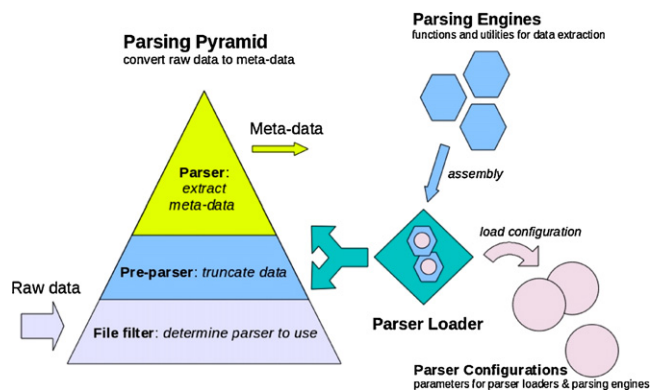


Fig. 2. Metadata parsing in CCDBT.

metadata using one configuration line written in the configuration file with the appropriate engines. Once a user drops a new configuration file into the configuration library folder, the new parser is created, and takes effect immediately. Users can also add, rewrite, and reload the parsing engines in a straight-forward manner, as the implementation is object-oriented. Experienced computational chemistry users can usually tell if a given file is a valid computational output, but programs are not likely to know this until they are taught (in the sense of machine learning) a set of selection rules. In order to extract useful metadata from a set of unknown files, the first action is to tell the program what type of computational output a file is likely to be (or not), the second task is to truncate the information-rich data to keep the part containing the metadata of interest only (often at the end of the file), and the third task is to actually extract the metadata. As shown in the “parsing pyramid” design in Fig. 2, the stream of raw data is fed into the base of the pyramid that is composed of parsers designed for uses at different levels, and the raw data is narrowed and filtered at each level of the pyramid to yield an extracted metadata stream as the output. All the parsers defined in the parser builder must have these items: (1) a parser should have a name associated with it, usually the name and version of a computational chemistry software; (2) it should be assigned to the default general loader or the customized loader inherited from the general parser loader base class; (3) a list of file suffixes used to filter the files of interest and a string pattern used to verify the match of the files; (4) it needs the suffix for the complementary output file paired with the main output file; (5) a parser should have an attribute that defines its priority when several parsers are available to parse a single file.

The last feature is a very useful attribute for treating large data sets in various formats. For example, assume that a general parser for MOLPRO [15] already exists in the parser library. However, if a newer build of the same software is released with the layout in the output changed, the old MOLPRO parser will not function properly. The CCDBT database application provides a solution to this problem by allowing the user to generate a new configuration file for the new version of MOLPRO with higher priority than that of the old MOLPRO parser. Also, sometimes a special type of calculation using the same software package will give an entirely different layout than the default layout in the output, and the user can use the same priority hierarchy to deal with such cases. The priority setting also enables a user to define a new parser with new variables such as those found in a locally modified software package or for new development versions of the computational chemistry software. Multiple parsers can be used without duplication of the metadata.

To some extent, the selection of metadata to be stored in the database is inspired by the Carnegie-Mellon Quantum Chemistry Archive (CMQCA) of the Pople group [16]. CMQCA is a collection of compressed results from the GAUSSIAN 80 program that could

CCDBT Metadata Query

Formula:

Advanced Search

Keywords Calculated By CodeVersion
 CalcType Method Basis Set
 Charge Multiplicity

Display Properties

☐ Title ☐ Keywords ☐ InputButGeom ☐ NumBasis ☐ NumFC ☐ InitGeom ☐ Point Group ☐ Elec State
☐ S2 ☐ Dipole ☐ Thermal ☐ Enthalpy ☐ Entropy ☐ Gibbs ☐ Freqs ☐ NumImagFreq
☒ CodeVersion ☐ CalcBy ☐ Job Status ☐ Convergence ☐ Fin Date ☐ CPUTime ☐ MemCost ☐ TimeCost ☒ FullPath
 Check all: ☐

Background Color ☐ white ☐ yellow ☐ mintcream ☐ palegreen ☐ lightyellow ☐ red ☒ black

Methods : b3lyp | CalcType : freq | You are searching for O%Ti%

1 - 10 of 290 entries

FinalGeom	Formula	Energy	ZPE	CalcType	Methods	Basis	Enthalpy	Gibbs	NImag	CodeVersion
 Jmol	O18Ti9(0,1)	-9000.5567802	0.0730088	FOpt; Freq	RB3LYP; RB3LYP	DGDZVP2; DGDZVP2	0.103362	0.016172	0	Gaussian 03, Revision E.01
 Jmol	O18Ti9(0,1)	-9000.5358867	0.0720889	FOpt; Freq	RB3LYP; RB3LYP	DGDZVP2; DGDZVP2	0.103494	0.014595	0	Gaussian 03, Revision E.01
 Jmol	O18Ti9(0,1)	-9000.5200946	0.071993	FOpt; Freq	RB3LYP; RB3LYP	DGDZVP2; DGDZVP2	0.10361	0.013224	0	Gaussian 03, Revision E.01
 Jmol	O18Ti9(0,1)	-9000.5003801	0.0713248	FOpt; Freq	RB3LYP; RB3LYP	DGDZVP2; DGDZVP2	0.102467	0.013063	1	Gaussian 03, Revision E.01
 Jmol	O18Ti9(0,1)	-9000.3978284	0.071672	FOpt; Freq	RB3LYP; RB3LYP	DGDZVP2; DGDZVP2	0.104487	0.001098	0	Gaussian 03, Revision E.01
 Jmol	O18Ti9(0,1)	-9000.0784698	0.0612627	Freq	RB3LYP	DGDZVP2	0.100807	-0.030337	2	Gaussian 03, Revision E.01
 Jmol	O16Ti8(0,1)	-8000.4740971	0.0642964	FOpt; Freq	RB3LYP; RB3LYP	DGDZVP2; DGDZVP2	0.091997	0.010006	0	Gaussian 03, Revision E.01
 Jmol	O16Ti8(0,1)	-8000.4310857	0.0656775	FOpt; Freq	RB3LYP; RB3LYP	DGDZVP2; DGDZVP2	0.091063	0.016306	0	Gaussian 03, Revision E.01
 Jmol	O16Ti8(0,1)	-8000.4015248	0.0630717	FOpt; Freq	RB3LYP; RB3LYP	DGDZVP2; DGDZVP2	0.090668	0.010541	0	Gaussian 03, Revision E.01
 Jmol	O16Ti8(0,1)	-8000.4005194	0.0629783	FOpt; Freq	RB3LYP; RB3LYP	DGDZVP2; DGDZVP2	0.091127	0.008813	0	Gaussian 03, Revision E.01

Fig. 3. A sample query in CCDBT in the web-based GUI.

Table 1
Parsed flags in the current CCDBT database system.^a

Flag	Type	Flag	Type	Flag	Type
StorageHost	varchar(40)	JobStatus	varchar(255)	OrbSym	longtext
FileRoot	varchar(255)	FinTime	varchar(255)	Dipole	varchar(255)
Account	varchar(40)	InitGeom	longtext	Freq	varchar(255)
FileRelativePath	varchar(255)	FinalGeom	longtext	AtomWeigh	varchar(255)
User	varchar(40)	PG	varchar(255)	Conditions	varchar(255)
GroupName	varchar(40)	ElecSym	varchar(255)	ReacGeom	longtext
LastModTime	date	NImag	smallint(6)	ProdGeom	longtext
ParsedBy	varchar(40)	Energy	double	MulCharge	double
RemotePath	varchar(950)	EnergyKcal	double	NatCharge	double
LocalPath	varchar(950)	ZPE	double	S2	double
Formula	varchar(255)	ZPEKcal	double	CodeVersion	varchar(255)
Charge	smallint(6)	HF	double	CalcMachine	varchar(255)
Multiplicity	smallint(6)	HFKcal	double	CalcBy	varchar(255)
Title	varchar(255)	Thermal	double	MemCost	varchar(255)
Keywords	varchar(255)	ThermalKcal	double	TimeCost	varchar(255)
CalcType	varchar(255)	Enthalpy	double	CPUTime	varchar(255)
Methods	varchar(255)	EnthalpyKcal	double	Convergence	varchar(255)
Basis	varchar(255)	Entropy	double	FullPath	varchar(950)
NumBasis	varchar(255)	EntropyKcal	double	InputButGeom	longtext
NumFC	varchar(255)	Gibbs	double	OtherInfo	longtext
NumVirt	varchar(255)	GibbsKcal	double	Comments	longtext

^a The length of the data field in bits is given in parenthesis in the "Type" column.

be accessed by the end user. Since this version of the software, an archive section following a similar standard is generated in the output of later versions of GAUSSIAN [17]. The golden rule of CMQCA is that the archived data must have self-reproducibility, i.e., one should be able to carry out the same calculation given the brief archive only. 2000 Hartree–Fock structures with STO-3G, 3-21G and 6-31G* are listed in the first version of CMQCA. Key information is stored following these rules: the input settings are echoed in

the archive output followed by the numerical results such as energies, dipoles, frequencies, and the final geometry is recorded if the calculation is searching for a local minimum or a saddle point. The metadata in the CCDBT database application shares many common features with CMQCA, and CMQCA was the prototype of the CCDBT metadata model.

Each metadata entry is extracted from a data file, and each metadata entry contains a number of record fields (columns in the

Table 2
The output of the MySQL query to search for all of the calculation files with the molecular formula containing C, H, N, O and Ru atoms.

Formula	Methods	Basis	CalcBy	Energy	Gibbs	Thermal	Enthalpy	FinTime
C ₁ H ₉ Al ₁ N ₂ O ₁₄ Ru ₁ Si ₃ (0,1)	RB3LYP; RB3LYP	GenECP; GenECP	xxxxxx; xxxxxx	−2413.081350	0.114581	0.200358	0.201302	26-Oct-2009; 27-Oct-2009
C ₁ H ₉ Al ₁ N ₂ O ₁₄ Ru ₁ Si ₃ (0,3)	UB3LYP; UB3LYP	GenECP; GenECP	xxxxxx; xxxxxx	−2413.054808	0.108401	0.198221	0.199166	29-Oct-2009; 9-Oct-2009
C ₂ H ₁₄ Al ₁ N ₂ O ₁₃ Ru ₁ Si ₃ (0,2)	UB3LYP; UB3LYP	GenECP; GenECP	xxxxxx; xxxxxx	−2378.914124	0.167983	0.258237	0.259181	25-Oct-2009; 26-Oct-2009
C ₂ H ₁₄ Al ₁ N ₂ O ₁₃ Ru ₁ Si ₃ (0,4)	UB3LYP; UB3LYP	GenECP; GenECP	xxxxxx; xxxxxx	−2378.897572	0.161490	0.257041	0.257986	05-Nov-2009; 05-Nov-2009
C ₂ H ₁₃ Al ₁ N ₂ O ₁₃ Ru ₁ Si ₃ (0,1)	RB3LYP; RB3LYP	GenECP; GenECP	xxxxxx; xxxxxx	−2378.328445	0.161085	0.247425	0.248369	22-Oct-2009; 22-Oct-2009
C ₂ H ₁₃ Al ₁ N ₂ O ₁₃ Ru ₁ Si ₃ (0,3)	UB3LYP; UB3LYP	GenECP; GenECP	xxxxxx; xxxxxx	−2378.311004	0.152835	0.245915	0.246859	01-Nov-2009; 01-Nov-2009
C ₁ H ₄ Al ₁ N ₂ O ₅ Ru ₁ (1+3)	UB3LYP; UB3LYP	GenECP; GenECP	yyyyyy; yyyyyy	−863.372406	0.030519	0.088050	0.088994	17-Sep-2009; 17-Sep-2009
C ₁ H ₄ Al ₁ N ₂ O ₅ Ru ₁ (1+3)	UB3LYP; UB3LYP	GenECP; GenECP	zzzzzz; zzzzzz	−863.372403	0.030524	0.088051	0.088995	21-Sep-2009; 21-Sep-2009
C ₁ H ₄ Al ₁ N ₂ O ₅ Ru ₁ (1+1)	RB3LYP; RB3LYP	GenECP; GenECP	zzzzzz; zzzzzz	−863.361828	0.033268	0.088511	0.089455	19-Sep-2009; 19-Sep-2009
C ₁ H ₄ Al ₁ N ₂ O ₅ Ru ₁ (1+1)	RB3LYP; RB3LYP	GenECP; GenECP	zzzzzz; zzzzzz	−863.330879	0.037255	0.088981	0.089925	13-Nov-2009; 13-Nov-2009
C ₁ H ₄ Al ₁ N ₂ O ₅ Ru ₁ (1+3)	UB3LYP; UB3LYP	GenECP; GenECP	zzzzzz; zzzzzz	−863.310040	0.032744	0.088019	0.088964	13-Nov-2009; 13-Nov-2009
C ₁ H ₄ Al ₁ N ₂ O ₅ Ru ₁ (1+3)	UB3LYP; UB3LYP	GenECP; GenECP	zzzzzz; zzzzzz	−863.310040	0.032744	0.088019	0.088964	13-Nov-2009; 13-Nov-2009
C ₁ H ₄ Al ₁ N ₂ O ₅ Ru ₁ (1+1)	RB3LYP; RB3LYP	GenECP; GenECP	zzzzzz; zzzzzz	−863.306125	0.032919	0.088271	0.089215	19-Sep-2009; 19-Sep-2009
C ₁ H ₄ Al ₁ N ₂ O ₅ Ru ₁ (1+3)	UB3LYP; UB3LYP	GenECP; GenECP	zzzzzz; zzzzzz	−863.305122	0.029928	0.087770	0.088714	25-Sep-2009; 25-Sep-2009
C ₁ H ₄ Al ₁ N ₂ O ₅ Ru ₁ (1+1)	RB3LYP; RB3LYP	GenECP; GenECP	zzzzzz; zzzzzz	−863.293988	0.032854	0.088214	0.089158	24-Sep-2009; 24-Sep-2009
C ₂ H ₉ Al ₁ N ₂ O ₄ Ru ₁ (1+2)	UB3LYP; UB3LYP	GenECP; GenECP	zzzzzz; zzzzzz	−829.252876	0.088630	0.145931	0.146875	19-Sep-2009; 19-Sep-2009
C ₂ H ₉ Al ₁ N ₂ O ₄ Ru ₁ (1+2)	UB3LYP; UB3LYP	GenECP; GenECP	zzzzzz; zzzzzz	−829.186030	0.090973	0.145309	0.146253	14-Nov-2009; 14-Nov-2009
C ₂ H ₉ Al ₁ N ₂ O ₄ Ru ₁ (1+2)	UB3LYP; UB3LYP	GenECP; GenECP	zzzzzz; zzzzzz	−829.179975	0.087600	0.145116	0.146060	25-Sep-2009; 25-Sep-2009
C ₂ H ₉ Al ₁ N ₂ O ₄ Ru ₁ (1+4)	UB3LYP; UB3LYP	GenECP; GenECP	zzzzzz; zzzzzz	−829.147777	0.083154	0.144308	0.145252	15-Nov-2009; 16-Nov-2009
C ₂ H ₉ Al ₁ N ₂ O ₄ Ru ₁ (1+4)	UB3LYP; UB3LYP	GenECP; GenECP	zzzzzz; zzzzzz	−829.145371	0.081975	0.144208	0.145152	01-Oct-2009; 01-Oct-2009
C ₂ H ₈ Al ₁ N ₂ O ₄ Ru ₁ (1+1)	RB3LYP; RB3LYP	GenECP; GenECP	zzzzzz; zzzzzz	−828.606920	0.077855	0.134924	0.135868	18-Sep-2009; 18-Sep-2009
C ₂ H ₈ Al ₁ N ₂ O ₄ Ru ₁ (1+1)	RB3LYP; RB3LYP	GenECP; GenECP	zzzzzz; zzzzzz	−828.584413	0.082359	0.135479	0.136424	13-Nov-2009; 13-Nov-2009
C ₂ H ₈ Al ₁ N ₂ O ₄ Ru ₁ (1+3)	UB3LYP; UB3LYP	GenECP; GenECP	zzzzzz; zzzzzz	−828.570402	0.078499	0.134402	0.135346	13-Nov-2009; 13-Nov-2009
C ₂ H ₈ Al ₁ N ₂ O ₄ Ru ₁ (1+3)	UB3LYP; UB3LYP	GenECP; GenECP	zzzzzz; zzzzzz	−828.569045	0.075227	0.134111	0.135055	26-Sep-2009; 26-Sep-2009
C ₂ H ₈ Al ₁ N ₂ O ₄ Ru ₁ (1+3)	UB3LYP; UB3LYP	GenECP; GenECP	zzzzzz; zzzzzz	−828.569044	0.075219	0.134109	0.135054	25-Sep-2009; 26-Sep-2009
C ₂ H ₈ Al ₁ N ₂ O ₄ Ru ₁ (1+1)	RB3LYP; RB3LYP	GenECP; GenECP	zzzzzz; zzzzzz	−828.536273	0.076967	0.134397	0.135341	24-Sep-2009; 25-Sep-2009
C ₂ H ₆ Al ₁ N ₂ O ₄ Ru ₁ (1+1)	RB3LYP; RB3LYP	GenECP; GenECP	zzzzzz; zzzzzz	−827.319534	0.056533	0.109707	0.110651	13-Nov-2009; 13-Nov-2009
C ₂ H ₆ Al ₁ N ₂ O ₄ Ru ₁ (1+3)	UB3LYP; UB3LYP	GenECP; GenECP	zzzzzz; zzzzzz	−827.318650	0.051287	0.108934	0.109878	15-Oct-2009; 15-Oct-2009
C ₂ H ₆ Al ₁ N ₂ O ₄ Ru ₁ (1+3)	UB3LYP; UB3LYP	GenECP; GenECP	zzzzzz; zzzzzz	−827.317129	0.053385	0.108897	0.109841	13-Nov-2009; 13-Nov-2009
C ₂ H ₆ Al ₁ N ₂ O ₄ Ru ₁ (1+1)	RB3LYP; RB3LYP	GenECP; GenECP	zzzzzz; zzzzzz	−827.305503	0.054258	0.109851	0.110795	16-Oct-2009; 16-Oct-2009
C ₄ H ₂₀ N ₁ O ₂ P ₃ Ru ₁ (0,1)	RB3LYP; RB3LYP	LANL2DZ; LANL2DZ	mmmm; mmmm	−482.978832	0.194819	0.260191	0.261135	26-Aug-2009; 26-Aug-2009
C ₄ H ₂₀ N ₁ O ₂ P ₃ Ru ₁ (0,1)	RB3LYP; RB3LYP	LANL2DZ; LANL2DZ	mmmm; mmmm	−482.971158	0.196181	0.260712	0.261656	25-Aug-2009; 25-Aug-2009
C ₄ H ₁₈ N ₁ O ₂ P ₃ Ru ₁ (0,1)	RB3LYP; RB3LYP	LANL2DZ; LANL2DZ	mmmm; mmmm	−481.786284	0.178125	0.240388	0.241332	25-Aug-2009; 25-Aug-2009
C ₄ H ₁₈ N ₁ O ₂ P ₃ Ru ₁ (0,1)	RB3LYP; RB3LYP	LANL2DZ; LANL2DZ	mmmm; mmmm	−481.781802	0.178650	0.240165	0.241110	25-Aug-2009; 25-Aug-2009

Adiabatic Bond Energy Analysis

Enter the Reactant's Formula to Search for the Adiabatic Bond Energies.

Formula:

Keywords CalcBy CodeVersion [gaussian]

CalcType [opt] Methods [b3lyp] Basis [DZVP2]

Charge [0] Erxn min [-999] Erxn max [999]

Clean All:

Background Color ☐ white ☐ yellow ☐ mintcream ☐ palegreen ☐ lightyellow ☐ red ☒ black

You are searching for H24B9N9

Basis : DGDZVP2 | Methods : b3lyp | CalcType : opt | Charge : 0 | Emin : -999.9 | Emax : 999.9 |

1 - 5 of 5 entries


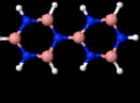


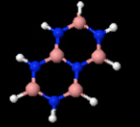
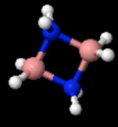


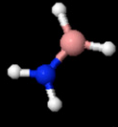


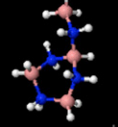

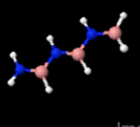
Reactant Formula	Reactant	Product1 Formula	Product1	Product2 Formula	Product2	Erxn(no zpe) kcal/mol
1 × H24B9N9(0,1)	 Jmol	1 × H10B6N6(0,1)	 Jmol	1 × H14B3N3(0,1)	 Jmol	11.0
1 × H24B9N9(0,1)	 Jmol	1 × H8B5N5(0,1)	 Jmol	2 × H8B2N2(0,1)	 Jmol	-2.5
1 × H24B9N9(0,1)	 Jmol	1 × H8B5N5(0,1)	 Jmol	4 × H4B1N1(0,1)	 Jmol	27.9
1 × H24B9N9(0,1)	 Jmol	1 × H8B5N5(0,1)	 Jmol	1 × H16B4N4(0,1)	 Jmol	-17.1
1 × H24B9N9(0,1)	 Jmol	3 × H8B3N3(0,1)	 Jmol	---	---	33.7

Fig. 4. An example data-mining application in CCDBT to analyze adiabatic bond dissociation energies of a given compound.

database table) that are filled with the parsed flags. The parsed flags in the initial CCDBT application are listed in Table 1. There are several types of parsed flags in the initial version. Most flags present the calculation settings and the calculated values of chemical properties from the output file (e.g., “Methods”, “ZPE”, and “Dipole”). Some flags provide additional information such as the calculation status and the location of the data file that cannot be directly extracted from the content of the output (e.g., “JobStatus” and “LastModTime”). The remaining flags are tagged by the account manager

to store the ownership and privilege of the metadata entries (e.g., “User”, “GroupName”, and “Account”).

The “RemotePath” flag (e.g., “account@server:relative_path”) is chosen to be the primary key in the CCDBT database table. In a database table, only one record entry with the same primary key value is allowed to exist, and this can thus be used to eliminate data redundancy in the database table. It also allows us to keep track of these calculations, so that they can be easily retrieved. Another important flag is the “Formula” flag given by the format of “formula

(charge, multiplicity)”, where the formula strings are defined in an order of elements so that carbon goes first, followed by hydrogen, and in alphabetical order thereafter. Every computational chemistry calculation has a molecular formula related to it, and one often distinguishes the calculations by the molecular formula alone. Therefore the “Formula” flag is a crucial parsed flag on which many of the metadata queries are based in the database application.

“During-transfer” mode and “on-demand scan” mode are the two data parsing modes that have been implemented. In the “during-transfer” mode, data are parsed instantly after they are successfully synchronized from the remote servers. New metadata are added to the database as soon as the newly produced computational outputs are transferred to the storage facility. In addition, working with the synchronization scheduler, this parsing mode enables the database to “grow” independent of any explicit user action. The “on-demand scan” mode, however, is designed for other type of needs, where the user would like to analyze and store fresh data before the routinely scheduled data synchronization and metadata parsing are performed, data outside the mirrored backup storage such as pre-existing local data need to be parsed, or data in the mirrored backup storage need to be re-parsed by new parsers.

2.4. User interface and data presentation

Both command-line and GUIs are implemented for the CCDBT program. The command-line interface has a few drawbacks. For example, it requires users to be familiar with the command line environment in Linux. More importantly, users are not able to easily follow the metadata meaningfully only in the graphical presentation, e.g., the XYZ coordinates for large molecules. However, the command line approach enables users to perform more sophisticated tasks than possible with a typical GUI. For example, the command-line interface can be directly integrated with a Linux shell environment, and can better interact with other applications available in that environment. One good example is that the database application can work with the task scheduler to do the data synchronization and parsing with different customized configurations. The web-based GUI, in contrast, is easier to use for a typical user, since the user does not need to memorize any of the commands and options. Data are also better presented in a webpage layout with enhanced visualization from the embedded JMOL applets. Another benefit of the web-based GUI is that hyperlinks in the displayed entries can provide quick access to the related data entries. The web-based GUI enables users to visually identify isomers of a chemical compounds with the same molecular formula that cannot be achieved by the command-line interface.

The command-line user interface has two modes, a menu-like interactive mode and a command-option mode. Under the interactive mode, menu entries for certain functionalities are printed on the screen, and users are prompted to select an entry from the menu to issue the corresponding command; the interactive mode is a simple alternative to the GUI. The command-option mode is executed from the scope of the operating system, where users call the executable followed by some specific options.

Queries in the database can be carried out by issuing a SQL command to the MySQL database, and can also be accomplished using the wrapper that allows users to submit simple queries and obtain responses without logging into MySQL. Table 2 shows the result of a sample query in our demo database. In this query, we are searching for “the calculation results with its molecule formula contains carbon (C), hydrogen (H), nitrogen (N), oxygen (O) and ruthenium (Ru) atoms” and print the calculation settings and the selected chemical properties (e.g., electronic energy, Gibbs free energy correction, enthalpy correction) on the screen, with the return records ordered by the electronic energy. This query was submitted to CCDBT and 35 entries of record were returned in less than 1 s on a Xeon 5300 Linux

workstation, with more than 100,000 metadata entries already in the database. The query is obtained by the following MySQL statement:

```
select Formula, Methods, Basis, CalcBy, Energy, Gibbs, Thermal, Enthalpy, FinTime from CCDBT_MAIN where Formula like "C%H%N%O%Ru%" and JobStatus like "CalcDone%" Order by Energy,
```

where CCDBT_MAIN is the name of the metadata table in MySQL and ‘%’ is the wildcard that matches a few characters. We are also able to use a wrapper to do the same thing by typing `ccdbtquery 'C*H*N*O*Ru*' in a Linux shell.`

Similar queries can be carried out using the web-based GUI, as shown in Fig. 3. In this example, 289 frequency calculation results of the molecules containing titanium (Ti) and oxygen (O) atoms using the B3LYP exchange correlation functional [18] are returned from the query ordered by their electronic energies, and the first 10 entries are displayed after pagination. Regular expressions are supported by our web-based search engine, which enable users to search for multiple formulae of interest at one time. For example, the user can look up all hydrocarbon compounds by search the pattern “C[0–9]+H[0–9]+”. The GUI is designed to have an appearance similar to any other advanced search engine that searches and filters the metadata using various criteria, so that most users will be able to use it without extensive training. The JMOL applets in the web-based GUI provide sufficient information of the molecular geometry in the data entry. The displayed molecules can be rotated, resized, and repositioned via mouse events, and different display styles can be chosen to meet different visualization needs.

Although the current version of the CCDBT provides information on data provenance in terms of the user, software package, computer, and electronic structure information, it does not guarantee that the calculation has been done correctly. It is still the responsibility of the CCDBT user to check the raw output file if needed. In future versions, we will enable the checking of the convergence of the geometry optimization, the convergence of the electronic energy, and the presence of imaginary frequencies by defining new terms in our parsers. In addition, the reaction energy tool described below can provide a quick check on the reliability of the total energy. Additional evaluations of the correctness of the calculation could be done by comparing energy and geometry parameters with appropriate systems but this would require substantial new work.

2.5. Metadata analysis

The CCDBT application is not only a useful tool to collect and display the calculation data, it is also a platform that allows user to perform higher level tasks such as data interpretation and data mining. The adiabatic bond dissociation energy analyzer shown in Fig. 4 is an example of such an application. Bond dissociation energies are important in evaluating whether a chemical reaction can occur or in investigating the stability of a molecule; thus it is a critical value that researchers may want to analyze. The calculation of bond dissociation energies (BDEs) is simple thermodynamics, but it can become tedious as the size of the reactant increases, as the reactant can be fragmented in various ways, and some of the meaningful reactions may be neglected. In addition, the calculation of BDEs can be quite complicated in terms of the number of terms in the electronic structure calculations that need to be included [19,20]. The adiabatic bond dissociation energy (ABDE) analyzer in the CCDBT web-based GUI tools is to help the researcher interpret his/her data. Given the molecular formula and other search criteria for the reactant, the ABDE analyzer searches for all of the related fragments in the database table containing the parsed metadata. The reaction energies of all the available $\text{Reactant} \rightarrow n\cdot\text{Product1} + m\cdot\text{Product2}$ reactions and $\text{Reactant} \rightarrow n\cdot\text{Product}$ reactions are calculated and displayed together with the molecular geometries of the reactants

and products. The hyperlinks in the display webpage are linked to the corresponding webpages to show the details of the reactant or each product. The benefit of this analyzer is that the reaction energies and the calculation details are woven together, rather than being isolated. The time cost to predict the BDE is greatly reduced, as it takes less than a few seconds for even very large molecules as the ABDE analyzer crawls over the database and returns the BDEs and the related metadata, which can consume substantial amounts of time if done by hand.

3. Progress, status, and outlook

The CCDBT application has been tested in our group for two years. Participants include undergraduate students, graduate students, and post-doctoral researchers. Almost all of the calculation files generated by the Dixon group in the last 6 years were processed by CCDBT. There are currently 49 users with a total number of 176 user accounts, all of which have been created and maintained by the account manager. 2.2 million files with a total disk volume of ~2.2 TB from various computing servers are stored on the storage facility and organized in tree-structured directories, with 193,000 metadata entries present in the database. These metadata were parsed by the available parsers in CCDBT, which are currently available for the NWChem [21,22], MOLPRO [15], and Gaussian [17] programs. These parsers were designed to parse multiple properties from the computational output files and reformat them into a uniform database structure. Although we have not yet demonstrated the application in a multiple-group environment, we do not see any issues with extending the software for such uses.

To summarize, we have built a molecular electronic property meta-database using CCDBT, from a large number of computational chemistry electronic structure output files in various formats from different remote computing clusters that are utilized by multiple users. The user needs only to provide the chemical formula to find information about a compound, and with the stored metadata, one can easily reproduce the calculations. A web-based GUI has been developed for data visualization and manipulation using JMOL for molecular structure display and property visualization. The application can be downloaded from <https://sourceforge.net/p/ccdbt>. We are also improving the documentation for CCDBT.

The CCDBT software can be readily extended to provide more parsing engines as well as more parsing functionalities. There are a number of features to be implemented in future versions of CCDBT. It will be possible to perform data mining and statistical analysis on the large metadata set. For example, it would be desirable to search for all unknown reactions with reaction energies in a certain range. Another example is the calculation of a specific thermodynamic property such as the proton affinity of all compounds in the database for which such a calculation is possible. These features would be of substantial value for comparing calculated data with standard data sets such as the G3/05 data set [23]. The database can enable searching for and identifying unexpected trends in large sets of computational data, for example, basis set convergence for different properties for groups of molecules.

Part of our objective is providing the ability to incorporate additional analysis components that can use the metadata in the CCDBT database. For example, we can develop a kinetics calculator that uses the thermodynamic data in the meta-database to generate input for transition state theory [24] or RRKM theory [25] calculations of rate constants, which would be similar to the currently available adiabatic bond dissociation energy analyzer. We could also calculate tunneling corrections to the rate constant using similar data such as the simple expression from Wigner [26] or those of Skodje and Truhlar [27]. Our goal is not to repeat features that are already available in systems such as AGUI [28] for displaying vibrational frequencies or intrinsic reaction coordinates or the

rate constant prediction tools in KHIMERA [29], both of which can directly read Gaussian output files. For these cases, our goal is to help the user find the necessary files for such software. We will also improve the tools and add-ons, for example, by providing more user-friendly features and options, e.g., column sorting and unit conversion.

Currently, the molecular formula in the CCDBT database is the major key for use in searching the database and filtering the results. Users can currently perform advanced structure searches using regular expressions, which can roughly and qualitatively meet the need to perform substructure queries. To fully utilize substructure queries, we plan to hash the stored geometry meta-data into InChI [1] or SMILES (Simplified Molecular Input Line Entry System) [30] string patterns which contains information about fragments and connectivity in a molecule, and to match string pattern of the substructure (usually a chemical group) with the string patterns pre-stored in the meta-database. Therefore users would need to only sketch a chemical group or a fragment in the molecules using a web interface to look up all the matched structures.

The CCDBT database application can handle most types of computational chemistry electronic structure output files which contain properties that can be concisely archived. The meta-data table in the database is expandable, so that the user can restructure the database to meet specific needs. For the output files that contain large amounts of data, such as those from molecular dynamics simulations or large scale potential energy surface scans, CCDBT can be made to store summary information for these calculations including the molecular formula, chemical structure, job settings, job status and a link to the output file. This will enable users to find the original data and to use other analysis tools. We are currently investigating the potential of CCDBT to capture critical data from Car-Parrinello ab initio molecular dynamics simulations [31], for example, vibrational information and structures and energies at extrema on the potential energy surface.

Moreover, the database application is not limited to the field of computational chemistry. It can be applied to other computational science fields by simply changing the motif of the database and the addition of new parsers.

Acknowledgements

This work was supported in part by funding from the United States Department of Energy, Office of Science, Advanced Scientific Computing Research, through a subcontract with the University of Tennessee. D.A. Dixon is indebted to the Robert Ramsay Endowment of the University of Alabama for partial support.

References

- [1] S.E. Stein, S.R. Heller, D. Tchekhovskoi, An open standard for chemical structure representation: The IUPAC Chemical Identifier, in: Proceedings of the 2003 International Chemical Information Conference (Nimes), Infonortics, 2003, pp. 131–143.
- [2] <http://www.synapticscience.com/seurat/database/>.
- [3] (a) H. Sui, R. Zhang, K. Pan, L. Luo, S. Chen, ChemDataBase – a database management system for facilitating data management in computational chemistry, in: 2008 International Multi-symposiums on Computer and Computational Sciences, 2008, pp. 190–193;
(b) L. Li, R. Zhang, J. Chen, Y. Zhang, L. Li, Z. Zhao, ChemDataBase 2: an enhanced chemical database management system for virtual screening, in: 2010 Fifth Annual ChinaGrid Conference (ChinaGrid), 2010, pp. 74–79.
- [4] G. Black, K.L. Schuchardt, D.K. Gracio, B. Palmer, The extensible computational chemistry environment: a problem solving environment for high performance theoretical chemistry, in: P.M.A. Sloot, D. Abramson, A.V. Bogdanov, J. Dongarra (Eds.), Computational Science – ICCS 2003, International Conference Saint Petersburg Russian Federation, Melbourne, Australia, Proceedings 2660, vol. 81, Springer Verlag, Berlin, Germany, 2003, pp. 122–131, G.D.
- [5] <http://ecce.emsl.pnl.gov/index.shtml>.
- [6] <http://www.primekinetics.org>.
- [7] (a) G. van Rossum, Python Tutorial, Technical Report CS-R9526, Centrum voor Wiskunde en Informatica (CWI), Amsterdam, 1995, May;

- (b) G. van Rossum, F.L. Drake, Python Reference Manual, PythonLabs, Virginia, USA, 2001, Python version 2.6 was used in our implementation <http://www.python.org>.
- [8] MySQL <http://www.mysql.com/>.
- [9] R. Pointer, paramiko: SSH2 protocol for Python. <http://www.lag.net/paramiko/>.
- [10] A. Dustman, mysql-python: MySQL for Python. <http://sourceforge.net/projects/mysql-python/>.
- [11] D.C. Lizenberger, PyCrypto: The Python Cryptography Toolkit. <http://www.dlitz.net/software/pycrypto/>.
- [12] J. Galbraith, J. Van Dyke, B. McClure, J. Bright, Secure Shell Public-Key Subsystem, IETF (internet draft), 2006, <http://tools.ietf.org>.
- [13] J. Galbraith, O. Saarenmaa, T. Ylonen, S. Lehtinen, SSH File Transfer Protocol (SFTP), IETF (internet draft), 2005, <http://www.ietf.org/internet-drafts/draft-ietf-secsh-filexfer-07.txt>.
- [14] Jmol: an open-source Java viewer for chemical structures in 3D. <http://jmol.sourceforge.net/>.
- [15] H.-J. Werner, P.J. Knowles, F.R. Manby, M. Schütz, P. Celani, G. Knizia, T. Korona, R. Lindh, A. Mitrushenkov, G. Rauhut, T.B. Adler, R.D. Amos, A. Bernhardsson, A. Berning, D.L. Cooper, M.J.O. Deegan, A.J. Dobbyn, F. Eckert, E. Goll, C. Hampel, A. Hesselmann, G. Hetzer, T. Hrenar, G. Jansen, C. Köppl, Y. Liu, A.W. Lloyd, R.A. Mata, A.J. May, S.J. McNicholas, W. Meyer, M.E. Mura, A. Nicklass, P. Palmieri, K. Pflüger, R. Pitzer, M. Reiher, T. Shiozaki, H. Stoll, A.J. Stone, R. Tarroni, T. Thorsteinsson, M. Wang, A. Wolf, MOLPRO, Version 2010.1, a Package of Ab initio Programs, 2010, <http://www.molpro.net>.
- [16] CMQCA, R.A. Whiteside, J.S. Binkley, R. Krishnan, D.J. DeFrees, H.B. Schlegel, J.A. Pople, Carnegie Mellon Quantum Chemistry Archive, Carnegie Mellon University, Pittsburgh, PA, 1981.
- [17] M.J. Frisch, G.W. Trucks, H.B. Schlegel, G.E. Scuseria, M.A. Robb, J.R. Cheeseman, G. Scalmani, V. Barone, B. Mennucci, G.A. Petersson, H. Nakatsuji, M. Caricato, X. Li, H.P. Hratchian, A.F. Izmaylov, J. Bloino, G. Zheng, J.L. Sonnenberg, M. Hada, M. Ehara, K. Toyota, R. Fukuda, J. Hasegawa, M. Ishida, T. Nakajima, Y. Honda, O. Kitao, H. Nakai, T. Vreven, J.A. Montgomery Jr., J.E. Peralta, F. Ogliaro, M. Bearpark, J.J. Heyd, E. Brothers, K.N. Kudin, V.N. Staroverov, R. Kobayashi, J. Normand, K. Raghavachari, A. Rendell, J.C. Burant, S.S. Iyengar, J. Tomasi, M. Cossi, N. Rega, N.J. Millam, M. Klene, J.E. Knox, J.B. Cross, V. Bakken, C. Adamo, J. Jaramillo, R. Gomperts, R.E. Stratmann, O. Yazyev, A.J. Austin, R. Cammi, C. Pomelli, J.W. Ochterski, R.L. Martin, K. Morokuma, V.G. Zakrzewski, G.A. Voth, P. Salvador, J.J. Dannenberg, S. Dapprich, A.D. Daniels, Ö. Farkas, J.B. Foresman, J.V. Ortiz, J. Cioslowski, D.J. Fox, Gaussian 09, Revision B.1, Gaussian, Inc., Wallingford, CT, 2009.
- [18] (a) A.D. Becke, Density-functional thermochemistry. III: the role of exact exchange, J. Chem. Phys. 98 (1993) 5648–5652;
(b) C. Lee, W. Yang, R.G. Parr, Development of the Colle–Salvetti correlation-energy formula into a functional of the electron density, Phys. Rev. B 37 (1988) 785–789.
- [19] D.J. Grant, M.H. Matus, J. Switzer, D.A. Dixon, J.S. Francisco, K.O. Christe, Bond dissociation energies in second row compounds, J. Phys. Chem. A 112 (2008) 3145–3156.
- [20] D.J. Grant, D.A. Dixon, Heats of formation and bond energies of the $H_{(3-n)}BX_n$ compounds for (X = F, Cl, Br, I, NH_2 , OH, and SH), J. Phys. Chem. A 113 (2009) 777–787.
- [21] E.J. Bylaska, W.A. de Jong, K. Kowalski, T.P. Straatsma, M. Valiev, D. Wang, E. Aprà, T.L. Windus, S. Hirata, M.T. Hackler, Y. Zhao, P.-D. Fan, R.J. Harrison, M. Dupuis, D.M.A. Smith, J. Nieplocha, V. Tipparaju, M. Krishnan, A.A. Auer, M. Nooijen, E. Brown, G. Cisneros, G.I. Fann, H. Früchtl, J. Garza, K. Hirao, R. Kendall, J.A. Nichols, K. Tsemekhman, K. Wolinski, J. Anchell, D. Bernholdt, P. Borowski, T. Clark, D. Clerc, H. Dachsel, M. Deegan, K. Dyall, D. Elwood, E. Glendening, M. Gutowski, A. Hess, J. Jaffe, B. Johnson, J. Ju, R. Kobayashi, R. Kutteh, Z. Lin, R. Littlefield, X. Long, B. Meng, T. Nakajima, S. Niu, L. Pollack, M. Rosing, G. Sandrone, M. Stave, H. Taylor, G. Thomas, J. van Lenthe, A. Wong, Z. Zhang, NWChem, A Computational Chemistry Package for Parallel Computers, Version 5.1, Pacific Northwest National Laboratory, Richland, Washington 99352-0999, USA, 2007.
- [22] M. Valiev, E.J. Bylaska, N. Govind, K. Kowalski, T.P. Straatsma, H.J.J. van Dam, D. Wang, J. Nieplocha, E. Aprà, T.L. Windus, W.A. de Jong, NWChem: a comprehensive and scalable open-source solution for large scale molecular simulations, Comput. Phys. Commun. 181 (2010) 1477–1489.
- [23] L.A. Curtiss, P.C. Redfern, K. Raghavachari, Assessment of Gaussian-3 and density-functional theories on the G3/05 test set of experimental energies, J. Chem. Phys. 123 (2005) 124107–124118.
- [24] J.I. Steinfeld, J.S. Francisco, W.L. Hase, Chemical Kinetics and Dynamics, 2nd edition, Prentice Hall, New Jersey, 1999.
- [25] K.A. Holbrook, M.J. Pilling, S.H. Robertson, Unimolecular Reactions, 2nd edition, Wiley, Chichester, UK, 1996.
- [26] E. Wigner, Crossing of potential thresholds in chemical reactions, Z. Phys. Chem. B 19 (1932) 203–207.
- [27] R.T. Skodje, D.G. Truhlar, Parabolic tunneling calculations, J. Phys. Chem. 85 (1981) 624–628.
- [28] <http://www.semichem.com/>.
- [29] KHiMERA, Version 3.2: A Software Tool for Calculations of Chemical Reactions Thermodynamics and Kinetics from First Principles, Kintech, Kinetic Technologies, Ltd., Moscow, 2003, <http://www.kintechlab.com/products/khimera/>.
- [30] <http://www.daylight.com/dayhtml/doc/theory/theory.smiles.html>;
<http://www.daylight.com/dayhtml.tutorials/languages/smiles/index.html>;
<http://cactus.nci.nih.gov/translate/>.
- [31] R. Car, M. Parrinello, Unified approach for molecular dynamics and density-functional theory, Phys. Rev. Lett. 55 (1985) 2471–2474.