Application Note

www.molinf.com

molecular informatics

DOI: 10.1002/minf.201400086

Multi-Server Approach for High-Throughput Molecular Descriptors Calculation based on Multi-Linear Algebraic Maps

César R. García-Jacas,*[a, b] Longendri Aguilera-Mendoza,[a] Reisel González-Pérez,[a] Yovani Marrero-Ponce,*[b, c, d] Liesner Acevedo-Martínez,[a] Stephen J. Barigye,[b, e] and Tatiana Avdeenko[f]

Abstract: The present report introduces a novel module of the QuBiLS-MIDAS software for the distributed computation of the 3D Multi-Linear algebraic molecular indices. The main motivation for developing this module is to deal with the computational complexity experienced during the calculation of the descriptors over large datasets. To accomplish this task, a multi-server computing platform named T-arenal was developed, which is suited for institutions with many workstations interconnected through a local network and without resources particularly destined for computation tasks. This new system was deployed in 337 workstations and it was perfectly integrated with the QuBiLS-MIDAS software. To illustrate the usability of the T-arenal platform, performance tests over a dataset comprised of 15 000 compounds are carried out, yielding a 52 and 60 fold reduction in the sequential processing time for the 2-Linear and 3-Linear indices, respectively. Therefore, it can be stated that the T-arenal based distribution of computation tasks constitutes a suitable strategy for performing high-throughput calculations of 3D Multi-Linear descriptors over thousands of chemical structures for posterior QSAR and/or ADME-Tox studies.

Keywords: TOMOCOMD-CARDD · QuBiLS-MIDAS · 3D N-linear algebraic descriptors · T-arenal · Platform of distributed tasks · Distributed computing system · Multi-server architecture

[a] *C. R. García-Jacas, L. Aguilera-Mendoza, R. González-Pérez, L. Acevedo-Martínez*
Grupo de Investigación de Bioinformática, Centro de Estudio de Matemática Computacional (CEMC), Universidad de las Ciencias Informáticas
La Habana, Cuba
**e-mail: crjacas@uci.cu*

[b] *C. R. García-Jacas, Y. Marrero-Ponce, S. J. Barigye*
Unit of Computer-Aided Molecular "Biosilico" Discovery and Bioinformatics Research (CAMD-BIR Unit), Faculty of Chemistry-Pharmacy, Universidad Central "Martha Abreu" de Las Villas
Santa Clara, 54830, Villa Clara, Cuba
**e-mail: ymponce@gmail.com*
ymarrero77@yahoo.es

[c] *Y. Marrero-Ponce*
Institut Universitari de Ciència Molecular, Universitat de València, Edifici d'Instituts de Paterna
P.O. Box 22085, E-46071, València, Spain

[d] *Y. Marrero-Ponce*
Grupo de Investigación en Estudios Químicos y Biológicos, Facultad de Ciencias Básicas,
Universidad Tecnológica de Bolívar
Cartagena de Indias, Bolívar, Colombia

[e] *S. J. Barigye*
Departamento de Química, Universidade Federal de Lavras, UFLA
Caixa Postal 3037, 37200-000 Lavras, MG, Brazil

[f] *T. Avdeenko*
Department of Economic Informatics, Novosibirsk State Technical University
Novosibirsk, Russia

Supporting Information for this article is available on the WWW under http://dx.doi.org/10.1002/minf.201400086.

1 Introduction

Chemoinformatics as a research field has among its objectives the codification of chemical information using computational methods, with the aim of accelerating the identification and/or optimization of compounds with desirable properties/activities.[1] To this end, a large number of logical/mathematical procedures that codify the molecular features or properties into numerical values, known as *molecular descriptors* (MDs), are commonly used.[2] Several software for the calculation of MDs have been developed, such as: DRAGON,[3] Mold2,[4] BlueDesc,[5] PowerMV,[6] PADEL[7] and others.[8] With the exception of PADEL, these programs do not employ parallelization strategies to compute their MDs and thus do not maximize the current computing architectures.

Recently, Marrero-Ponce et al. introduced the QuBiLS-MIDAS [Quadratic, Bilinear and N-Linear Maps based on N-tuple Spatial Metric Similarity Matrices and Atomic Weightings] software,[9] which computes geometric (3D) alignment-free MDs for non-covalent relations among *N* atoms of a molecule and are collectively denominated as *3D N-linear MDs* (also see SI1).[10] This program has remarkable features such as: multi-core processing, batch processing mode, data cleaning module, allows the custom configuration of the MDs, fully cross-platform, and others (see Table 4 in García-Jacas et al.[9] for a complete comparison) which grant the QuBiLS-MIDAS software several advantages

over other programs reported. This software is freely available via INTERNET at: http://tomocomd.com/.

The *3D 2-linear MDs*, as a subfamily of the *3D N-linear MDs*, are employed when non-covalent relations between two atoms are taken into account. These relations are codified using several similarity metrics (e.g. Manhattan, Euclidean, Canberra and others) and the obtained values are represented in the *two-tuple spatial-similarity matrix*.[10b] Moreover, and as a natural extension of the aforementioned subfamily, the *3D 3-linear* and *4-linear MDs* were defined to consider relations among three and four atoms, respectively, using multi-metrics, e.g. *bond angle* to relate three atoms and *dihedral angle* to relate four atoms. This information is condensed into *three-tuple* and *four-tuple spatial-similarity matrices*, respectively. These approaches were used for the first time in the definition of 3D-MDs.[10a]

The *3D N-linear MDs* have been assessed in three different chemoinformatics studies: 1) variability analysis based on Shannon's Entropy, which evaluates the ability of discriminating compounds with different chemical features,[11] 2) linear independence of the captured information using the Principal Component Analysis technique,[12] and 3) QSAR studies. The first and second study demonstrated that the *3D N-Linear MDs* present comparable-to-superior variability and codify orthogonal information with regard to 3D DRAGON MDs.[10] In the third study, the obtained results were superior to those reported according to the *cross-validation leave-one-out* (Q^2_{loo}) and *external* (Q^2_{ext}) *predictive ability* parameters for the considered datasets,[13] with these results further corroborated using non-parametric statistical procedures (see SI2).

As was previously mentioned, one of the main features of the QuBiLS-MIDAS software is its multi-core processing capability. However, it is not enough to satisfy the computational demand for the high-throughput calculation of *3D N-linear MDs* over large datasets. For example, in a previous study[9] it was observed that the computation of 20 280 *3D 4-linear MDs* for the **PrimScreen1** dataset comprised by 1000 compounds, took approximately 106 370 seconds (29 hours) employing 16 processing threads (8 native cores with multi-threading each one). As can be noted, despite the fact that the processing time is considerably less than when one processor (639 006 seconds) is used, it still is quite high. In fact, if the dataset size is increased by *m* molecules, then the processing time will be raised by approximately 106**m* times. Thus, other computational alternatives should be taken into consideration to enable quicker computations of these MDs.

Nowadays, distributed computing is a popular method of dealing with huge calculations. In this sense, there are systems known as Desktop Grids or Volunteer Computing Projects, such as BOINC,[14] OurGrid,[15] Hadoop,[16] Java Heterogeneous Distributed Computing (JHDC)[17] and others,[18] based on the principle of using the computational resources available across networks to process computationally large problems.[19] All these systems support Bag-of-Tasks (BoT) processing, that is, each task is divided into independent sub-tasks which are processed by the processing nodes linked to the system. So, the high-throughput computation of *3D N-linear MDs* on thousands of structures could be implemented as *BoT calculations*, where each subtask will be comprised by a subset of molecules and the MDs to compute it.

To accomplish this, the JHDC system was employed because of the following reasons: 1) it's fully cross-platform without recompilation of the source code; 2) it presents a modular design which allows that each component to be isolated and thus independently developed, modified and tested; 3) it simultaneously supports several computation tasks of diverse complexity; 4) it's developed in Java language and thus the integration with the library for calculating *3D N-linear MDs* is simple;[9] 5) it's open source; 6) it's of general purpose and has been successfully employed in Bio-Chem-informatics' applications;[20] and 7) it's designed to work in an INTRANET environment.

However, as almost all distributed computing platforms, the JHDC system is based on *client/server architecture* (CSA) and thus it is not always suitable to support a huge number of connected workstations, whereby, the use of multiple servers could be a suited alternative. In this sense, a Multi-Tiered Distributed Computing Platform was proposed.[21] This platform consists of an *n-ary tree of nodes* where the *internal nodes* work as schedulers and the *leaf nodes* perform the processing. However, an appropriate deployment and use of this could be complex due to: 1) there is no *a priori* knowledge on the width and depth of the tree in order to ensure a good behavior; 2) the processing nodes must be statically configured to one of the many scheduling nodes and thus, it is difficult to perform an initial configuration of the system; and 3) the recursive splitting of the tasks for Bio-Chem-informatics applications cannot be indicated for all programs because some of these are platform-dependent (e.g. computation of MDs with Mold2 program). Therefore, this paper unveils strategies to address these challenges, by creating an environment through which the QuBiLS-MIDAS software is capable of performing high-throughput distributed calculations of the *3D N-linear MDs* over large datasets employing a multi-server approach.

2 Overview of the Multi-Server Distributed Computing System

The distributed computing system described in this report, denominated as *T-arenal*, is created with the purpose of using a large number of computational resources under its control in a local network, without limitations in the size of the system. To this end, a multi-server approach was implemented by combining the *peer-to-peer* and *client-server* architectures within one model, in such way that the servers can work together without losing computational power.[22]

T-arenal is divided in two main parts: back-end and front-end. The front-end is the means through which the users can access to the functionalities on the system, while the back-end is responsible of performing all requests made via the front-end. The front-end can either be the desktop graphical interface provided with T-arenal (see Subsection 2.3), or any other interface developed. The Java RMI (Remote Method Invoke) communication technology[23] for message passing among the components of T-arenal was employed, as well as Java Sockets or Apache FTP[24] to exchange large data files.

2.1 Back-End. Multi-Server Approach

The back-end of T-arenal is based on a multi-server model organized as a tree of three levels (see Figure 1). It is divided into three software components: *root server*, *request server* and *client*. The *root server* constitutes the link from the back-end to the front-end and it is thus responsible for handling or assigning to one of the *request servers* the demands performed via the front-end. In addition, it allocates an incoming calculation task to the most suitable *request server* for its processing (see SI3) and determines when a *request server* must *collaborate* with another to perform a task. Moreover, the *root server* has a mechanism of allocating the *clients* of the system to the corresponding *request servers* according to user-defined allocating rules. So, the deployment of T-arenal is simple because the *clients* always establish communication with the *root server* in order to obtain the configuration (IP address and port) of the *request server* to operate with.

On the other hand, the *request server* is responsible of breaking down the assigned task(s) into smaller sub-tasks (work units) according to a user-defined algorithm, and collecting the results obtained from the computations performed by the *clients* in order to construct the final solution. Another functionality of a *request server* is to control the work units created but yet to be processed. The amount of *request servers* to use in the system is according to the logical structure in which the workstations are grouped (e.g. per sub-network). There again, new *request servers* can be dynamically added or removed without affecting the distributed system.

It is important to remark that the *request server*(s) can handle several tasks of different priorities at the same time. This number of tasks can be user-defined (by default is equal to 1) and its minimum value is 0. This lower bound indicates that all resources belonging to a *request server* will be used to *collaborate* in the accomplishment of a task placed in another *request server*. This *collaboration schema* (see SI3.1) is also used when the number of tasks is less than the number of available *request servers*. The *request server* that is *collaborating* in a task works like a proxy between their *clients* and the helped *request server*. Thus, the high concurrence present in a system based on CSA due to the many connections is improved. In this sense, if there are *m request servers* each one with *n clients* and only one *request server* is working, this will receive as maximum

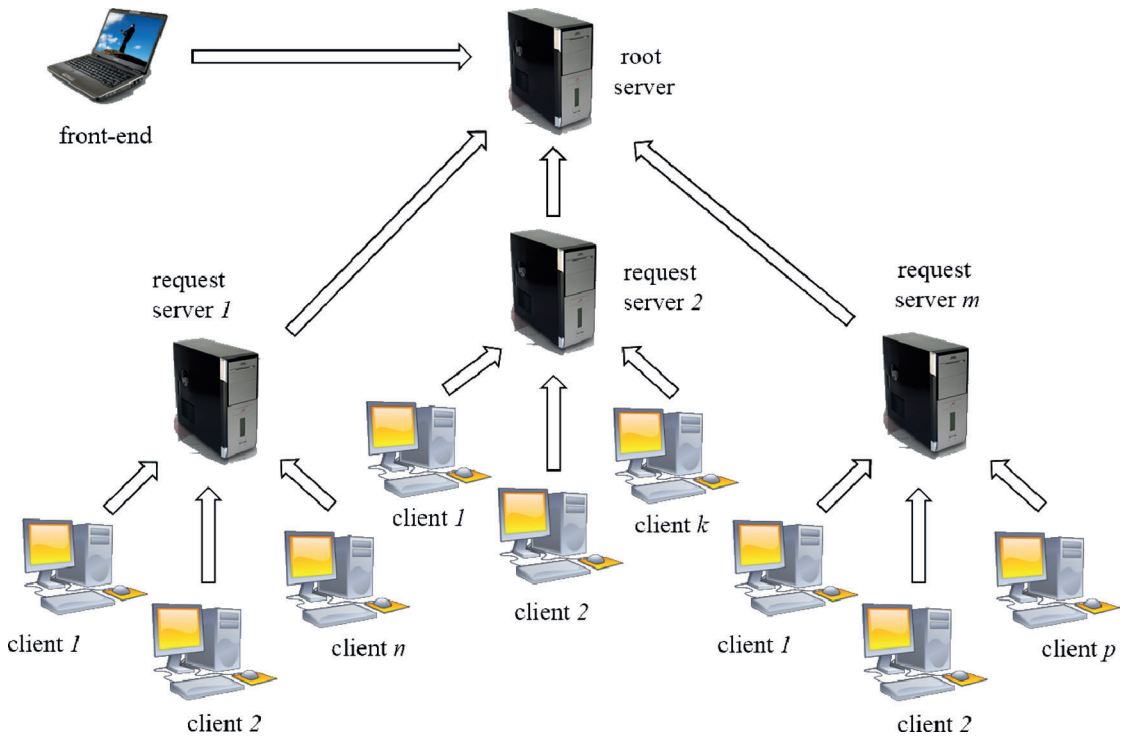


Figure 1. General architecture of the T-arenal system.

$(m-1)+n$ requests which are much less than if all existing *clients* ($m*n$) carried out requests to the same server.

Finally, the *client* module is responsible of processing the work units assigned from the corresponding *request server*. To this end, the *client* frequently requests a work unit, performs its processing, returns the result to the *request server* and once again requests another work unit. If at any time the *client* does not receive a work unit to process, it will go into a "sleep mode" for some period before performing the previous operations. Moreover, if a work unit is assigned, then the *client* will not contact again the corresponding *request server* until the sub-task is completed or an exception occurs. All the communications between a *request server* and their *clients* are initiated by the latter. If the *client* loses the communication or is disabled from working with the corresponding *request server*, then it connects again to the *root server* to obtain the configuration of another *request server*. Scheme 1 is a flowchart illustrating how a calculation task is processed in the T-arenal system.

2.2 T-Arenal Library

A library was added to T-arenal in order to programmatically interact with the system. This library completely conceals the technical details, topology and communication protocol of the computing system from the developers and it thus facilitates the development of custom applications that require the T-arenal system. There is a *SystemConnection* class that constitutes the starting point to establish communication with the system. To build an instance of this class, it is necessary to pass as parameters the information of the *final user* (username and password) and *root server* (IP address and communication port). If the *SystemConnection's* object is successfully created, then the *final user* is authenticated into the system and can access to all corresponding functionalities.

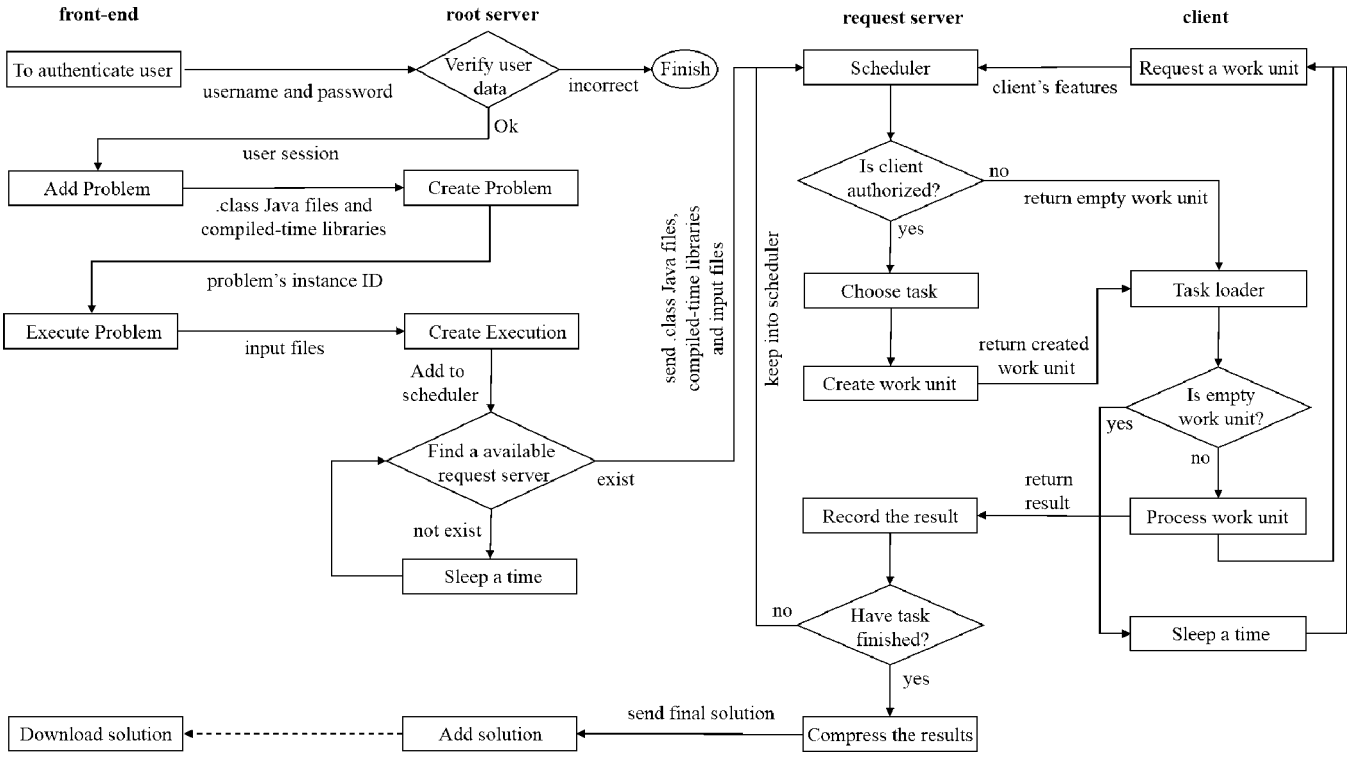
2.3 Front-End. T-Arenal Desktop Graphic User Interface

The T-arenal Desktop Graphic User Interface constitutes the front-end provided with the T-arenal platform. This standalone application connects and disconnects from the *root server* without affecting the system's operation. Its development was performed using the API previously described and its objective is to provide access to all functionalities implemented on T-arenal. For instance, the owner of a particular task can monitor its progress, as well as download the results compressed into a single Zip archive. For more information see T-arenal User Manual freely available at: http://tomocomd.com/.

3 Distributed QuBiLS-MIDAS Software

3.1 Distributed Computation Task of N-Linear Algebraic Molecular Descriptors.

To assess the utility of the multi-server approach, the *3D N-linear MDs*[10] were implemented to be calculated over T-



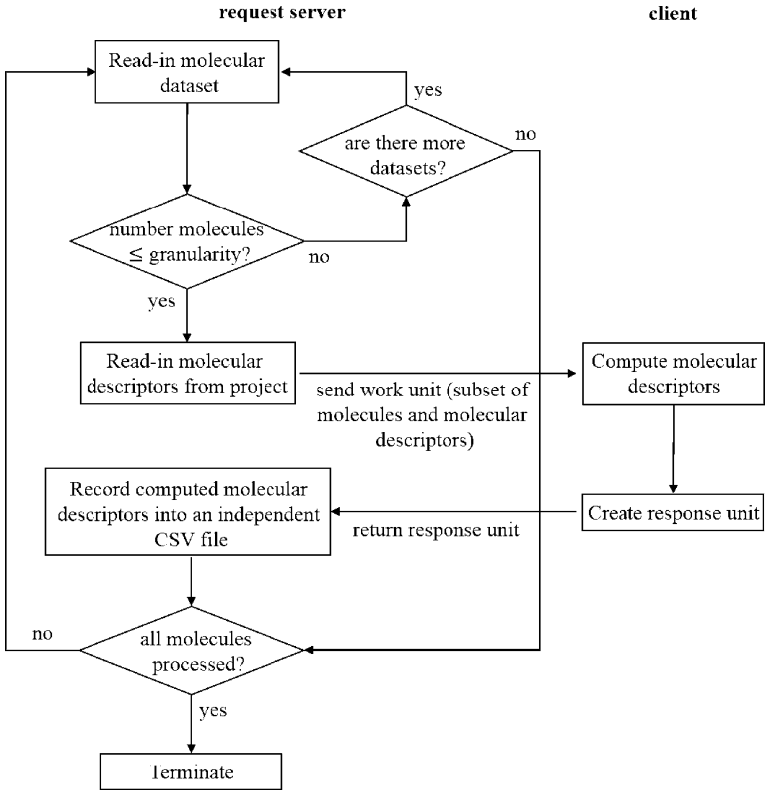
Scheme 1. General workflow of the T-arenal system to perform a calculation task allocated in a *request server*.

arenal. To this end, the programming library explained in SI4 was employed. In SI5 the UML diagram of the classes responsible of performing the distributed calculation is shown, where *QuBiLsDataManager* and *QuBiLsTask* constitute the two main classes of the application. An instance of the former receives as input one or several datasets in MDL MOL/SDF formats, and in addition one or several projects previously saved with the QuBiLS-MIDAS software.[9] In the overridden *generateWorkUnit* method used to build the sub-tasks, each *dataset/project pair* is taken into consideration and for each of these a subset of molecules and the corresponding *3D N-Linear MDs* are obtained, which together constitute a work unit. The subset of molecules is according to the user-defined *granularity* in the configuration file provided with this application, with 1 being the minimum value. It is important to remark that the *granularity* is according to the computational complexity of MDs to be computed, and the features of the workstations connected to T-arenal. With a suitable *granularity* overload in the interconnection network is avoided because of constant communication of the *clients* with the corresponding *request server*.

Once the work units are assigned, then in each *client* an instance of the *QuBiLsTask* class is built to perform the processing. The results obtained are sent back to the *QuBiLsDataManager* instance through the overridden *processResults* method. These should be saved in one CSV (Comma Separated Values) output file in the same order that the corresponding work units were created. However, the results may not necessarily be received in the aforementioned order due to the fact that T-arenal is designed to work in a dynamic environment. Therefore, to avoid this situation the obtained results are saved as independent CSV files. The distributed calculation finalizes when the *3D N-Linear MDs* are computed for all compounds (see Scheme 2).

3.2 Distributed Computation Module for the QuBiLS-MIDAS Software

To run this distributed application on the T-arenal system it is mandatory to upload the *.class files* (compiled Java files) corresponding to the classes represented in SI5, as well as the compiled-time libraries employed. To this end, a module for interacting with the distributed system was developed in the QuBiLS-MIDAS software. In this way, it is only necessary to select the option that indicates that the *3D N-Linear MDs* will be computed on T-arenal (see Figure 2A). Subsequently, a dialog window is shown in order for the user authenticate himself on the system (see Figure 2B), and if this process is successfully performed then another dialog window for setting up the parameters of the calculation is shown as well. Once the task is configured, a *problem* named *"DistributedQuBiLS"* is automatically created if it is not yet recorded on T-arenal. Next, from this *problem* a particular distributed computation is initialized



Scheme 2. General workflow of the 3D N-Linear molecular descriptors distributed computation.

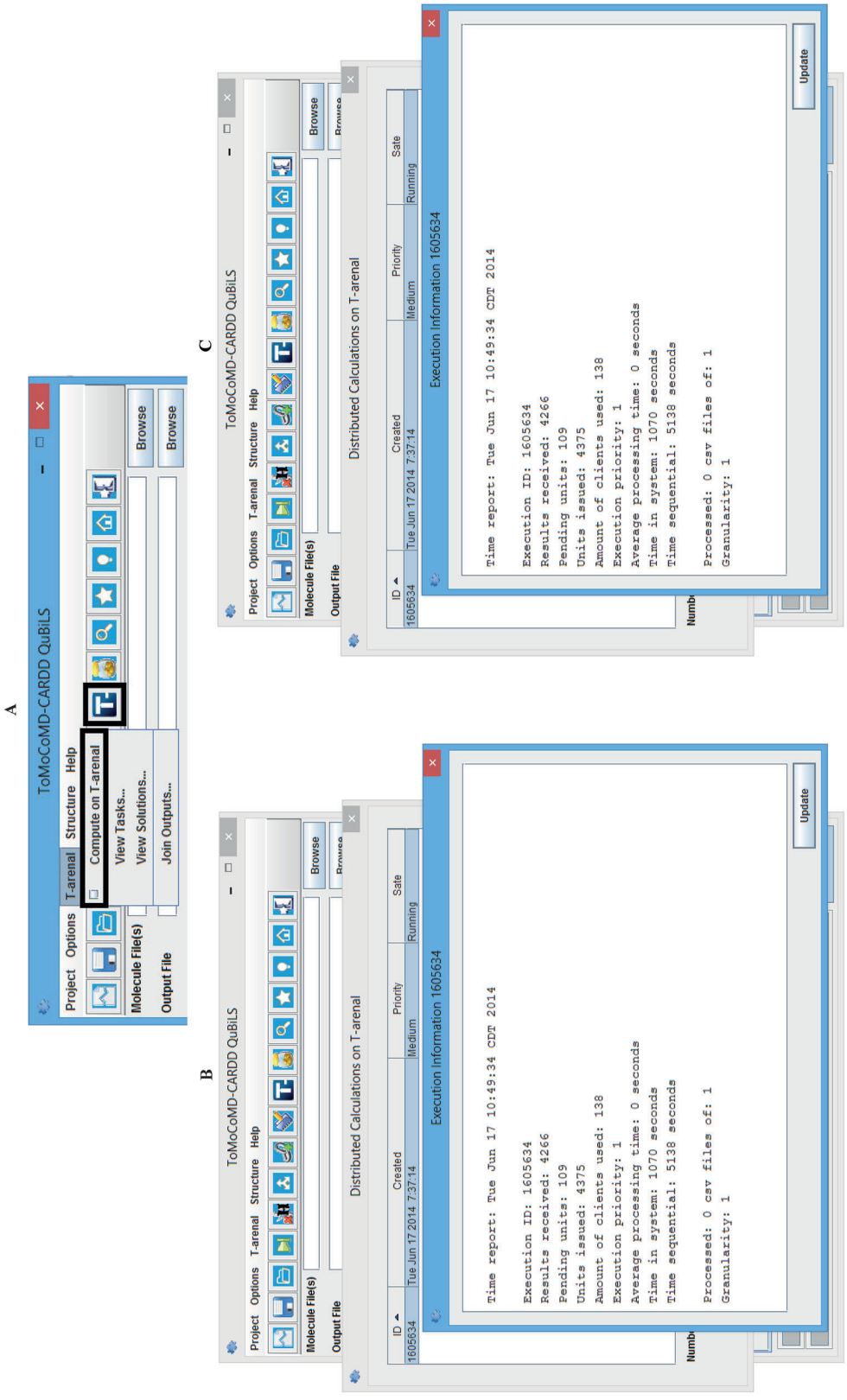


Figure 2. A) Illustration on how the T-arenal system option may be accessed from the desktop Graphic User Interface in the new module of the QuBiLS-MIDAS software. Note that T-arenal system may also be accessed from the toolbar (see highlighted T- icon) B) Dialog window to login into T-arenal system. C) User interface where the progress of a distributed computation process running on T-arenal system is monitored.

with the transference of the datasets and the MDs chosen by user. Finally, the user can close the QuBiLS-MIDAS software or perform another calculation either locally on the workstation or on the T-arenal platform.

Also, this new module permits the user to monitor (see Figure 2C) or stop the execution of a determined computation, as well as to download or delete from the system the obtained solutions. In addition, an option to join the results belonging to the work units created during the processing into a unique output file, is provided in the T-arenal menu. Lastly, it is important to highlight that the distributed computations of MDs may also be started from the Batch Processing module belonging to the QuBiLS-MIDAS software.

3.3 Distributed Performance Test

In order to perform the distributed computation of the *3D N-Linear MDs*, the T-arenal system through a *ghost image* (a clone disk image used for deploying programs in large number of workstations simultaneously) was deployed in 337 workstations belonging to the computing laboratories of the University of Informatics Sciences, Havana, Cuba. T-arenal was configured with a *root server* and three *request servers* with an equal number of *clients* (workstations). Each server have a 2.20 GHz Core(TM)2 Duo E4500 with 1 GB of RAM and 160 GB of hard disk space. The *clients* present a wide variety of hardware features (see Table 1), have Nova GNU/Linux (https://humanos.uci.cu/nova/) as operating system and are connected to T-arenal by means of a non-dedicated network Ethernet LAN to 100 Mb/s.

For evaluating the performance of this distributed application the *speedup* and *efficiency* metrics were taken into consideration. The former is computed by dividing the *"best" sequential time* of the algorithm by the parallel time obtained with p processors (workstations), while the latter is calculated by dividing the *speedup* by the corresponding number of processors used. The *speedup* metric indicates to what extent a parallel algorithm improves its sequential version, with the maximum *speedup* value being equal to the number of processors employed to accomplish the algorithm. The *efficiency* metric represents the effectiveness (a value between 0–1) with which an algorithm utilizes the p processors assigned. These metrics are generally computed for algorithms executed on dedicated-systems of homogenous architectures. However, because of the heterogeneity of the computational resources linked to the T-arenal system (see Table 1), the *"best" sequential time* was determined on the workstation with the best features in the set of computers: an Intel(R) Core(TM)2 Duo CPU E4500 2.20 GHz with 2 GB RAM.

For assessing the performance of the application, 12480 and 7488 *2-linear* and *3-linear* MDs were computed, respectively, on the **PrimScreen15** dataset comprised by 15000 structures (http://www.otavachemicals.com/-download-compound-libraries/cat_view/110-diversity-sets/133-primscreen-15). It is important to highlight that as T-arenal is deployed in a dynamic environment, then the number of workstations used to compute the *speedup* metric cannot be fixed *a priori*. In this sense, the authorized *clients* (may not necessarily be switched on) to perform work requests are gradually increased to achieve a greater number of computational resources working on the calculations. The granularity used was of 6 and 2 molecules (values obtained after several internal tests) for the *2-linear* and *3-linear MDs*, respectively.

Table 2 shows the processing time (per molecule and descriptor), speedup and efficiency attained during the computation of the MDs. The processing time is also graphically represented in the Figure 3. As can be observed, the computing time always decreases as the amount of processors (*clients* or workstations) used in the computation is increased. So, it can be observed that the sequential calculation time is reduced from 49349 seconds (13 hours) to 950 seconds (16 minutes) and from 166017 seconds (46 hours) to 2783 seconds (46 minutes) in the computation of the *3D 2-linear* and *3-linear MDs*, using as maximum number of *clients* 265 and 282, respectively.

On the other hand, when the behavior of the *speedup* is analyzed, it is observed that this is not proportional to the number of processors used. Similar behavior is observed in the assessment of the *efficiency* attained with the computational resources utilized. However, both behaviors may be due to the fact that the maximum number of clients employed is not constant throughout the calculations because some of them could crash (e.g. clients could be switched off). In addition, the *speedup* and *efficiency* metrics were calculated taking into account the *"best" sequential time*, determined under a Core(TM)2 Duo architecture. However,

Table 1. Hardware features and the corresponding number and percentages calculated with respect to the 337 workstations linked to T-arenal system.

Processors			RAM		
Type	Amount	Percent (%)	Size (MB)	Amount	Percent (%)
Pentium(R) 4	224	66.47	≤ 512	69	20.47
Celeron(R)	72	21.36	> 512 and ≤ 768	37	10.97
Core(TM)2 Duo	25	7.41	> 768 and ≤ 1024	217	64.39
Pentium(R) Dual-Core	14	3.26	> 1024 and ≤ 1536	5	1.48
Pentium(R) D	1	0.30	> 1536	9	2.67
Atom(TM)	1	0.30			

Table 2. Distributed calculation results of the duplex (2-linear) and ternary (3-linear) QuBiLS-MIDAS MDs.

Number of Processors	Processing time (s)	Speedup	Efficiency	Processing time for one molecule (s)	Processing time for one descriptor (s)
Duplex QuBiLS-MIDAS indices					
1	49349	1.000	1.000	3.290	3.954
27	6534	7.553	0.280	0.436	0.524
45	3697	13.348	0.297	0.246	0.296
71	2220	22.229	0.313	0.148	0.178
115	1571	31.412	0.273	0.105	0.126
157	1443	34.199	0.218	0.096	0.116
187	1238	39.862	0.213	0.083	0.099
265	950	51.946	0.196	0.063	0.076
Ternary QuBiLS-MIDAS indices					
1	166017	1.000	1.000	11.068	22.171
66	11773	14.102	0.214	0.785	1.572
93	8639	19.217	0.207	0.576	1.154
161	7339	22.621	0.141	0.489	0.980
179	6531	25.420	0.142	0.435	0.872
184	5346	31.054	0.169	0.356	0.714
190	4362	38.060	0.200	0.291	0.583
282	2783	59.654	0.212	0.186	0.372

the majority of the workstations have poorer hardware features (87.83 % are Pentium(R) 4 and Celeron(R), see Table 1) than those from which the *"best" sequential time* was computed and thus, these workstations have more probability of being available to perform the calculations. Consequently, the processing time is greater than if the workstations to be used were mainly based on Core(TM)2 Duo architecture.

Despite the good performance achieved, it is important to point out that the distributed calculation of the *3D N-Linear MDs* is not always advantageous. This is due to the fact that sometimes the calculation of the MDs is quicker on a single workstation than on a wide distributed computing system, particularly when a small number of compounds are to be studied and/or few indices are to be computed. Therefore, the developed module for the

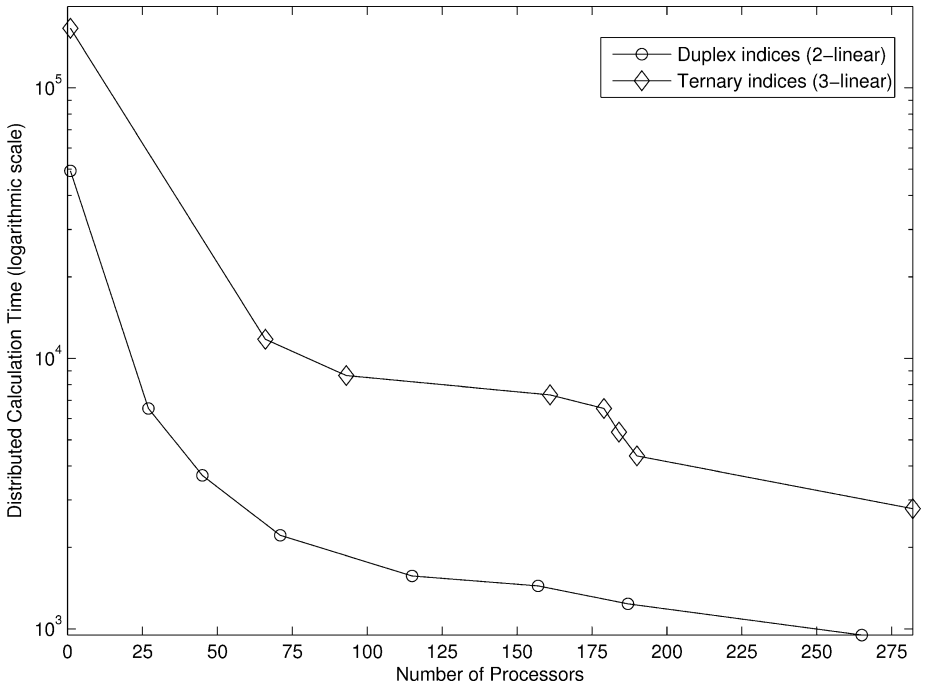


Figure 3. Graphic representation of the processing time achieved during the distributed calculation of the Duplex and Ternary algebraic molecular descriptors.

QuBiLS-MIDAS program is recommended when thousands of compounds and/or MDs are considered.

4 Conclusions

In this manuscript a novel module belonging to the QuBiLS-MIDAS software for the distributed calculation of *3D N-Linear MDs* was presented. To this end, a new distributed computing platform, denominated as T-arenal, was developed by using a multi-server approach to offer an alternative for high-throughput calculation tasks. T-arenal in a dynamic environment constituted of 337 heterogeneous workstations was deployed, and for its evaluation 12480 *2-linear* and 7488 *3-linear MDs* were computed over a dataset comprised of 15000 compounds. The obtained results demonstrate that the sequential processing time of the considered MDs is reduced by 52 times in the case of the former and 60 for the latter, when their computation is distributed among various workstations. Therefore, it can be concluded that the novel module constitutes a valuable tool to perform high-throughput calculations of *3D N-linear MDs* over large datasets, and in this way contribute to the faster characterization of compounds when QSAR or ADME-Tox studies are to be carried out.

5 Future Outlooks

Currently, a MPI-based parallel library is being developed to provide a suitable tool for using the computational power available in the super-computers. In addition, another version of the QuBiLS-MIDAS software based on the Graphic Processing Units (GPUs) is being implemented as well.

Supplementary Information Available

The current version of the QuBiLS-MIDAS software, the distributed computing system developed in this report and the respective user manuals are freely available in the ToMoCoMD Framework web site (http://tomocomd.com/). In addition, the scheduling strategy and programming library to developed distributed applications for T-arenal system are provided.

Acknowledgement

Marrero-Ponce, Y. thanks to the program 'International Professor' for a fellowship to work at *Cartagena University* in 2013–2014.

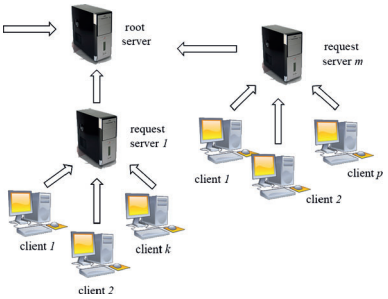
References

[1] a) F. K. Brown, *Annu. Rep. Med. Chem.* **1998**, *33*, 375–384; b) T. Engel, *J. Chem. Inf. Comput. Sci.* **2006**, *46*, 2267–2277.
[2] R. Todeschini, V. Consonni, *Molecular Descriptors for Chemoinformatics*, Vol. 1, 1edst edWILEY-VCH, Weinheim, **2009**.
[3] *DRAGON*, v 6.0, Milano Chemometrics and QSAR Research Group, Milano, Italy, **2010**.
[4] H. Hong, Q. Xie, W. Ge, F. Qian, H. Fang, L. Shi, Z. Su, R. Perkins, W. Tong, *J. Chem. Inf. Comput. Sci.* **2008**, *48*, 1337–1344.
[5] *BlueDesk*, University of Tübingen, Tübingen, Germany, **2008**.
[6] K. Liu, J. Feng, S. S. Young, *J. Chem. Inf. Model.* **2005**, *45*, 515–522.
[7] C. W. Yap, *J. Comput. Chem.* **2011**, *32*, 1466–1474.
[8] a) *CODESSA III*, Semichen, Shawnee, USA; b) *ADRIANA.Code*, Molecular Networks GmbH, Erlangen, Germany, c) *MODESLAB*, v1.5, MODesLab.com, **2002**; d) *Molconn-Z*, v4.10, Hall Associates Consulting – Molconn.com, Quincy, MA, USA; e) R. Guha, *CDK Descriptor Calculator GUI*, v1.3.9.
[9] C. R. García-Jacas, Y. Marrero-Ponce, L. Acevedo-Martínez, S. J. Barigye, J. R. Valdés-Martiní, E. Contreras-Torres, *J. Comput. Chem.* **2014**, *35*, 1395–1409.
[10] a) C. R. García-Jacas, Y. Marrero-Ponce, S. J. Barigye, J. R. Valdés-Martiní, O. M. Rivera-Borroto, J. O. Verbel, *Curr. Drug Metab.* **2014**, *15*, 441–469; b) Y. Marrero-Ponce, C. R. García-Jacas, S. J. Barigye, J. R. Valdés-Martiní, O. M. Rivera-Borroto, R. W. Pino-Urias, N. Cubillán, Y. J. Alvarado, *Curr. Bioinf.* **2014**, in press.
[11] J. W. Godden, F. L. Stahura, J. Bajorath, *J. Chem. Inf. Comput. Sci.* **2000**, *40*, 796–800.
[12] K. V. Mardia, J. T. Kent, J. M. Bibby, *Multivariate Analysis*, Academic Press, London, **1979**.
[13] a) J. J. Sutherland, L. A. O'Brien, D. F. Weaver, *J. Med. Chem.* **2004**, *47*, 5541–5554; b) F. Bonachéra, D. Horvath, *J. Chem. Inf. Model.* **2008**, *48*, 409–425; c) P. Tosco, T. Balle, *J. Chem. Inf. Model.* **2011**, *52*, 302–307; d) G. Hinselmann, L. Rosenbaum, A. Jahn, N. Fechner, A. Zell, *J. Cheminf.* **2011**, *3*, 3; e) A. Klamt, M. Thormann, K. Wichmann, P. Tosco, *J. Chem. Inf. Model.* **2012**, *52*, 2157–2164.
[14] a) D. P. Anderson, in *Proc. Fifth IEEE/ACM Int. Workshop on Grid Computing, IEEE*, **2004**, pp. 4–10; b) D. P. Anderson, E. Korpela, R. Walton, in *First Int. Conf. e-Science and Grid Computing, IEEE*, Melbourne, Vic., **2005**, pp. 203–211.
[15] W. Cirne, F. Brasileiro, N. Andrade, L. Costa, A. Andrade, R. Novaes, M. Mowbray, *J Grid Comp.* **2006**, *4*, 225–246.
[16] C. Lam, *Hadoop in Action*, Manning Publications Co., Greenwich, CT, USA, **2010**.
[17] a) T. Keane, R. Allen, T. J. Naughton, J. McInerney, J. Waldron, in *Scientific Engineering for Distributed Java Applications*, Vol. 2604 (Eds: N. Guelfi, E. Astesiano, G. Reggio), Springer, Heidelberg, **2003**, pp. 122–131; b) T. Keane, Thesis, National University of Ireland Maynooth **2004**.
[18] a) A. Marosi, G. Gombas, Z. Balaton, P. Kacsuk, T. Kiss, in *Making Grids Work*, Springer, US, **2008**, pp. 365–376; b) M. J. Litzkow, M. Livny, M. W. Mutka, *8th Int. Conf. Distributed Comp. Syst. IEEE*, San Jose, CA, **1988**, pp. 104–111; c) M. Neary, A. Phipps, S. Richman, P. Cappello, in *Euro-Par 2000 Parallel Processing*, Vol. 1900 (Eds: A. Bode, T. Ludwig, W. Karl, R. Wismüller), Springer, Heidelberg, **2000**, pp. 1231–1238; d) eG. Fedak, C. Germain, V. Neri, F. Cappello, in *Proc. First IEEE/ACM Int. Symp. Cluster Computing and the Grid, IEEE*, Brisbane, Qld., **2001**, pp. 582–587.
[19] a) P. Domingues, P. Marques, L. Silva, *Int. Conf. Workshops on Parallel Processing, IEEE*, **2005**, pp. 469–476; b) D. Kondo, M.

Taufer, C. Brooks, H. Casanova, A. Chien, *Proc. 18th Int. Symp. Parallel Distributed Processing, IEEE*, **2004**, p. 26.
[20] a) T. M. Keane, T. J. Naughton, *Bioinformatics* **2005**, *21*, 1705 – 1706; b) T. M. Keane, T. J. Naughton, S. A. Travers, J. O. McInerney, G. McCormack, *Bioinformatics* **2005**, *20*, 969 – 974.
[21] A. Page, T. Keane, R. Allen, T. J. Naughton, J. Waldron, *Proc. 2nd Int. Conf. Principles and Practice of Programming in Java*, Computer Science Press, Inc., Kilkenny City, Ireland, **2003**, pp. 191 – 194.
[22] M. Livny, M. Melman, *SIGMETRICS Perform. Eval. Rev.* **1982**, *11*, 47 – 55.
[23] E. Pitt, K. McNiff, *Java.rmi: The Remote Method Invocation Guide*, Addison-Wesley Longman, Boston, MA, USA, **2001**.
[24] The Apache Software Foundation.

Received: June 22, 2014
Accepted: September 17, 2014
Published online: ■■ ▮▮, 0000

ESSAY



*C. R. García-Jacas,**
L. Aguilera-Mendoza,
*R. González-Pérez, Y. Marrero-Ponce,**
L. Acevedo-Martínez, S. J. Barigye,
T. Avdeenko

■■ – ■■

Multi-Server Approach for High-Throughput Molecular Descriptors Calculation based on Multi-Linear Algebraic Maps