CONCURRENCY AND COMPUTATION: PRACTICE AND EXPERIENCE
Concurrency Computat.: Pract. Exper. 2014; **26**:1787–1798
Published online 14 August 2013 in Wiley Online Library (wileyonlinelibrary.com). DOI: 10.1002/cpe.3117

SPECIAL ISSUE PAPER

A performance/cost model for a CUDA drug discovery application on physical and public cloud infrastructures

Ginés D. Guerrero [1,*,†,‡], Richard M. Wallace [2,‡], José L. Vázquez-Poletti [2], José M. Cecilia [3], José M. García [1], Daniel Mozos [2] and Horacio Pérez-Sánchez [3]

[1] *Dept. of Computer Architecture, University of Murcia, 30080, Murcia, Spain*
[2] *Dept. of Computer Architecture and Automation, Complutense University of Madrid, 28040, Madrid, Spain*
[3] *Dept. of Computer Science, Catholic University of Murcia, 30107, Murcia, Spain*

SUMMARY

Virtual Screening (VS) methods can considerably aid drug discovery research, predicting how ligands interact with drug targets. *BINDSURF* is an efficient and fast blind VS methodology for the determination of protein binding sites, depending on the ligand, using the massively parallel architecture of graphics processing units(GPUs) for fast unbiased prescreening of large ligand databases. In this contribution, we provide a performance/cost model for the execution of this application on both local system and public cloud infrastructures. With our model, it is possible to determine which is the best infrastructure to use in terms of execution time and costs for any given problem to be solved by BINDSURF. Conclusions obtained from our study can be extrapolated to other GPU-based VS methodologies. Copyright © 2013 John Wiley & Sons, Ltd.

Received 28 February 2013; Revised 9 July 2013; Accepted 19 July 2013

KEY WORDS: cloud computing; high performance computing; CUDA; energy efficiency; drug discovery; virtual screening

1. INTRODUCTION

In clinical research, it is crucial to determine the safety and effectiveness of current drugs and to accelerate findings in basic research – such as discovery of new leads and active compounds – into meaningful health outcomes. Both objectives require processing large data sets of protein structures that are available in biological databases such as Protein Data Bank (PDB) [1] and also derived from genomic data using techniques such as homology modeling [2]. Screenings in lab and compound optimization are expensive and slow methods. Bioinformatics can vastly help clinical research work toward the goals of safety and effectiveness by providing predictions of toxicity of drugs and activity in non-tested targets and by evolving discovered active compounds into drugs for clinical trials.

These goals can be achieved thanks to the availability of bioinformatics tools and virtual screening (VS) methods that allow testing of all required hypothesis before clinical trials. Current VS methods, such as docking, fail to make good toxicity and activity predictions due to being constrained by computational resources; even the most efficient VS methods cannot process large biological databases in a reasonable time frame. Thus, these constraints impose serious limitations in many areas of biomedical research dependent on computational drug discovery.

*Correspondence to: Ginés D. Guerrero, Dept. of Computer Architecture, University of Murcia, 30080, Murcia, Spain.
†E-mail: gines.guerrero@ditec.um.es
‡These authors contributed equally.

The use of massively parallel hardware architectures, such as graphics processing units (GPUs), can tremendously overcome these limitations. GPUs have become increasingly popular in the high performance computing area by combining impressive computational power with the demanding requirements of real-time graphics for the lucrative mass market of the gaming industry [3]. Scientists have exploited this power in arguably every computational domain, and the GPU has emerged as a key resource in applications where parallelism is the common denominator [4]. To maintain this momentum, new hardware features have been progressively added by NVIDIA (Santa Clara, CA, USA) to their range of GPUs, with the Kepler architecture [5] being the most recent milestone in this path.

Large system clusters are adopting the use of these relatively inexpensive and powerful devices as a way of accelerating computationally intensive parts of the applications. One of the current fastest supercomputers, Titan, located at the DOE Oak Ridge National Laboratory in Tennessee, USA [6], is equipped with AMD Opteron Processors and the latest generation of NVIDIA K20x GPUs. Current GPUs have a great impact on the power consumption of the system, as a high-end GPU may well increase the power consumption of a cluster node up to 30%. This is a critical concern especially for very large datacenters, where the cost dedicated to supply power to such computers represents an important fraction of the total cost of ownership [7].

Reducing power consumption in these large installations is now becoming an urgent concern as several governments (e.g., USA and UK) are creating taxes targeting facilities that consume too much electricity [8, 9]. For instance, some of the more well-known datacenters on the Internet, such as Google and Facebook among others, consumed about 0.5% of the overall electricity in the world during 2005. When electricity needed for cooling and power distribution is also considered that number increases up to 1% [10]. The research community is also aware of this issue and it is making efforts in developing reduced-power installations. For instance, the GREEN500 list [11] shows the 500 most power efficient computers in the world. In this way, we can see a clear shift from the traditional metric fLoating point operations per second (FLOPS) to FLOPS per watt.

Virtualization techniques may provide significant energy savings as they enable greater resource usage through sharing hardware resources among several users thus reducing the required amount of instances of that particular device. As a result, virtualization is increasingly being adopted in datacenters. In particular, cloud computing is an inherently energy-efficient virtualization technique [12], where services run remotely in a ubiquitous computing cloud providing scalable and virtualized resources. Thus, peak loads can be moved to other parts of the cloud, and the aggregation of a cloud's resources can provide higher hardware use [13]. Public cloud providers offer their services in a 'pay-as-you-go' fashion and provide an alternative to local system infrastructures. This alternative to local system infrastructures only becomes real for a large data amounts and long execution times.

In this work, we analyze this scope of computation by targeting a GPU-based VS method called *BINDSURF* [14]. We propose a performance/cost model for this application allowing the user to decide which infrastructure – be it a local one or that of a well-known public cloud provider – is optimal for a given problem type and size. The execution of a GPU intensive application such as BINDSURF [15] may strain an Institution's budget when processing great amounts of data. The greater the number of computational physical resources or greater execution time for those resources, the more the total cost is increased, even for unused local infrastructures.

The rest of the paper is organized as follows. Section 2 briefly introduces the preliminary knowledge to better understand the rest of the article. Section 3 explains the experiments performed for crafting the model that is formulated in Section 4. Section 5 shows the model in action using realistic conditions, and finally, the paper ends with some conclusions and directions for future work.

2. RELATED WORK

2.1. Bioinformatics approaches for drug discovery

In discovering new leads, compound optimization, toxicity evaluation, and additional stages of the drug discovery process, VS methods screen large databases of molecules to find ones that fit an established criteria [16]. Among the many available VS methods for this purpose, we decided to

DOI: 10.1002/cpe

use protein–ligand docking [17, 18]. Docking simulations are typically carried out on the protein surface using known methods such Autodock [19], Glide [20] and DOCK [21]. The docking region is commonly derived from the position of a particular ligand in the protein-ligand complex or from the crystal structure of the protein without any ligand. The main problem of many docking methods is to make the assumption that once the binding site is specified, all ligands will interact with the protein in the same region, completely discarding other areas of the protein.

BINDSURF [15] overcomes this problem by dividing the whole protein surface into arbitrary, independent regions (aka 'spots') and using GPUs to process these spots in parallel. Thus, a large ligand database is screened against the target protein over its whole surface simultaneously with docking simulations for each ligand performed simultaneously for all specified protein spots. This results in new spots found after examining the distribution of scoring function values over the entire protein surface.

2.2. *Exploitation of HPC resources for bioinformatics applications*

High performance computing (HPC) platforms are attractive for technical computation due to their ability to produce data parallel solutions and reduce the makespan needed to simulate biological and chemical processes. Moderately sized, tightly coupled applications can be hosted on large-scale supercomputing systems. These moderately sized applications are good candidates for cloud computing. Even with the increased performance of desk-side systems, there is still a need for these applications to scale [22]. Typical bioinformatics applications are scientific work-flows composed of programs or services based on known and accepted methods and algorithms [23, 24]. Given this, unless applications are written for parallel execution, taking advantage of cloud or HPC systems efficiently will be infeasible [25]. In such cases, applications should use parallelism techniques, such as data fragmentation [26, 27].

Several approaches for parallelizing bioinformatics applications exist based on grid solutions [25, 28, 29]. CloudBLAST [30] uses the MapReduce paradigm to parallelize bioinformatics tools. Another BLAST execution, AzureBlast [31] uses 'split/join' patterns. BlastReduce [32] is a parallel read mapping algorithm using Hadoop [33]. Using Hadoop and MapReduce [33, 34], Biodoop [35] as a bioinformatics applications suite provides a general-purpose parallelization technology that successfully handles distributed bioinformatics problems. EvolvingSpace [36] is a data-centric system for integrating bioinformatics applications. Condor [37], while not a cloud implementation is a distributed execution engine for applications. In [38] the MapReduce-MPI library successfully executes BLAST and SOM in parallel. MapReduce [34] is particularly well adapted to run bioinformatics applications. A study [39] shows that it is common to have execution of large numbers of independent tasks or tasks that perform minimal inter-task communication in parallel for the bioinformatics domain.

2.3. *Statistics of cloud computing*

The promise of cloud computing is delivering all the functionality of existing information technology services and new functionalities that were previously infeasible as it dramatically reduces the up-front costs of computing that deter many organizations from deploying many cutting-edge IT services [40]. This scale of cost for computing coupled with data centers operating at 10% to 30% of their available computing power and desktop computers at less than 5% calls into question asset and power costs. Equally important are maintenance and service costs that are a steady drain on corporate resources. A recent survey by Gartner research indicated that about two-thirds of the average corporate IT staffing budget goes toward routine support and maintenance activities [41].

Cloud computing gives researchers advantages from the convergence of computational resource efficiency, where the power of modern computers are used more efficiently through highly scalable hardware and software resources and the ability to have these resources available on an as-needed basis with rapid deployment, parallel batch processing, use of compute-intensive applications, and interactive applications that respond in real time to user requirements [42].

By employing cloud computing, operational cost savings for energy, and keeping service level agreements improve large-scale computing acceptability with greater environmental sustainability

DOI: 10.1002/cpe

[43]. In the case of Amazon, estimates by Hamilton from Amazon Services [44] show the cost and operation based on a three-year amortization schedule (the low end of the industry nominal schedule of replacement every 3 to 7 years) account for 53% of the budget. An additional 42% of the energy costs include direct power consumption (approximately 19%) and cooling infrastructure (23%) amortized over a 15-year period [45]. In this way, the comparison of the total cost of ownership between cloud infrastructures and local infrastructures has been recently studied. For instance, Kashef and Altmann [46] suggest a cost model for hybrid clouds. Strebel and Stage [47] proposed an economic decision model for business software application. Truong and Dustdar [48] presented several techniques to estimate costs for several traditional scientific applications.

3. EXPERIMENTS

3.1. Experiment definition

We carried out VS calculations using BINDSURF for the direct prediction of binding poses using three different ligands that conveniently represent chemical diversity of large compound databases. They will be referred to as ligands A, B, and C. Ligand A is a blood clotting cofactor recently discovered by us [49]. Ligand B and ligand C have been extracted from their protein data bank complexes with the respective IDS $2byr$ and $3p4w$. In the docking calculations, we accounted for different numbers of Monte Carlo steps such as 5, 10, 50, 500, 5000, and 50,000. An optimal value for the *steps* parameter does not exist for all different ligand types (A, B, and C). Therefore, it is convenient to perform VS calculations using different values of this parameter as we might be interested in short simulations ($steps = 5, 10, 50$) to obtain qualitative information about potential hot spots in the surface screening approach for millions of different ligands. In other situations, we are more interested in obtaining accurate predictions for a smaller set of ligands using higher values for the *steps* parameter such as 500, 5000, and 50, 000. The outcome of a docking simulation performed by BINDSURF for a type A ligand (PDB identifier $1qcf$) is shown in Figure 1.

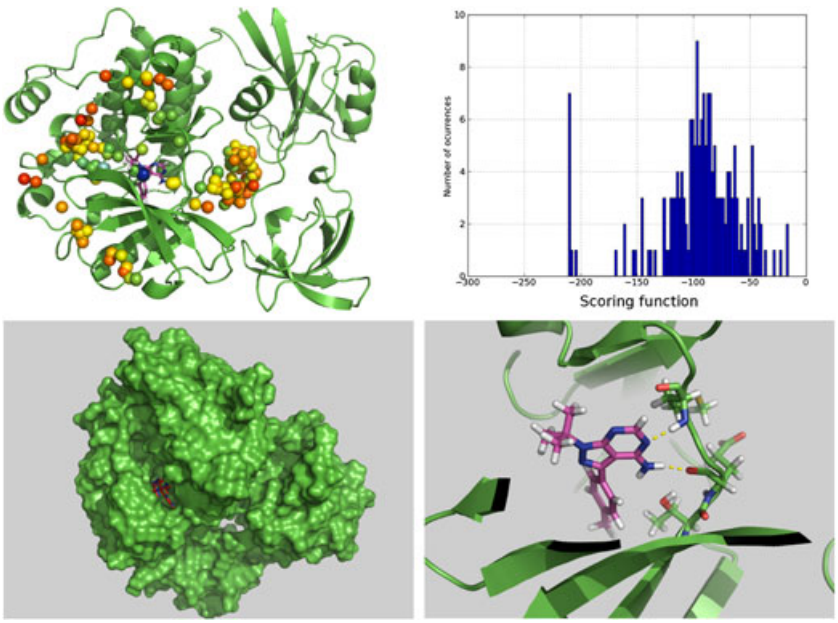


Figure 1. Surface screening results for PDB:1QCF. From up left to down right: (a) beads represent protein spots, and the color of each bead is related with the value of the scoring function, so colors from red to blue indicate lower values for the scoring function; (b) histogram with the distribution of scoring function values; (c) red and blue molecules represent crystallographic and predicted pose for the ligand, root-mean-square deviation is lower than 1 Angstrom; and (d) depiction of the hydrogen bonds established by the ligand with the closest residues.

DOI: 10.1002/cpe

Table I. Platforms system specifications.

(a) Local machine	
Processor:	Intel Xeon E5620@2.4 Ghz
Memory:	16 GB
2xGPU NVIDIA Tesla C2050	
GPU:	GF100
Memory size:	3072 MB
Memory bandwidth:	144 GB/s
Stream processors:	448
Max power draw:	238 W
(b) Amazon EC2	
Processor:	2xIntel Xeon X5570@2.93 GHz
Memory:	22 GB
2xGPU NVIDIA Tesla M2050	
GPU:	GF100
Memory size:	3072 MB
Memory bandwidth:	148.4 GB/s
Stream processors:	448
Max power draw:	225 W

3.2. Infrastructure used

The local architecture used to perform our experiments is shown in Table Ia. The system is composed of an Intel Xeon E5620 CPU with 4 cores running at 2.4 GHz, 16 GB of RAM memory, and two NVIDIA Tesla C2050 graphics cards.

The cloud infrastructure is one offered by Amazon through its Elastic Compute Cloud services (EC2)[§]. As BINDSURF is coded using CUDA, the Cluster GPU instances were the only possible choice. The specifications of the GPU provided by Amazon EC2 are shown in Table Ib. Being a public cloud provider, Amazon charges per hour of use. Each 'Quadruple Extra Large' instance (the one providing GPUs) deployed on the US East Region costs \$2.1/h[¶].

Even though both targeted platforms provide two GPUs on each machine, the experiments were conducted using just one in order to avoid contention issues.

3.3. Single experiment results

In this Section, we show our experiments performed on both local and cloud infrastructures before we develop the local and cloud models. These experiments are based on 10 executions of BINDSURF per ligand, varying the number of Monte Carlo steps. Table II shows the execution time (in minutes) of BINDSURF on both infrastructures for each ligand type. Execution times for the cloud infrastructure are higher than the local infrastructure with ligand type B (2byr) having the lowest values and ligand type C (3p4w), the highest for both infrastructures. As expected, the local infrastructure is faster than the cloud infrastructure mainly due to the low number of steps and the overhead introduced by virtualization and communications for the cloud infrastructure.

Power consumption for the local machine is shown in Table III. These are averaged values for each set of experiments. The power consumption of the local machine is mainly driven by the execution time and GPU usage. Processing ligand type C, for instance, requires higher preprocessing time than the rest of the ligands, while energy consumption for pre-processing is lower than the energy consumption in the processing phase.

[§]http://aws.amazon.com/ec2/.
[¶]http://aws.amazon.com/ec2/pricing/.

DOI: 10.1002/cpe

Table II. Time in minutes when processing different types of ligands in a single machine with different Monte carlo steps.

	Ligand type A		Ligand type B		Ligand type C	
Steps	Local	Cloud	Local	Cloud	Local	Cloud
5	0.77	1.46	0.66	1.16	0.97	1.75
10	0.77	1.48	0.66	1.17	0.98	1.98
50	0.85	1.56	0.74	1.26	1.05	1.89
500	1.90	2.60	1.83	2.33	2.02	2.86
5000	15.53	16.18	14.71	15.18	14.80	15.60

Table III. Watt-hour power consumption for one ligand of each type with different number of simulation steps executing on a local machine.

	Ligand type A	Ligand type B	Ligand type C
5	286	289	281
10	286	290	280
50	292	295	283
500	332	333	317
5000	366	363	357

4. EXECUTION MODEL

This section describes the performance/cost model defined using the results obtained in the previous section. The model is categorized by infrastructure (local and cloud) and then by ligand type (l1c4, 2byr, and 3p4w).

The model predicts the behavior of BINDSURF when more machines are added to the resource pool. As the followed workload distribution is very simple, no transfers have been considered because the total number of ligands to be processed is divided between the available machines from the beginning.

4.1. Local model

The individual execution time, expressed in minutes, for each ligand of the local model is given by three equations shown in 1 resulting from fitting the results from Section 3.3.

$$\begin{aligned} t_{local_A} &= 10^{-8} s_A^2 + 0.0029 s_A + 0.6699 \\ t_{local_B} &= 2 \cdot 10^{-9} s_B^2 + 0.0028 s_B + 0.5858 \\ t_{local_C} &= 8 \cdot 10^{-9} s_C^2 + 0.0027 s_C + 0.8765 \end{aligned} \tag{1}$$

where s_x is the number of simulation steps for processing a given ligand type (i.e., A, B, or C).

Equation 2 shows the extrapolated total execution time.

$$T_{local_x} = \frac{t_{local_x} \cdot l_x}{m} \tag{2}$$

where t_{local_x} is the time obtained for a given ligand x in Equation 1. The number of processed ligands is represented by l_x and the number of physical machines by m.

Local costs can be expressed by Equation 3:

$$C_{local_x} = C_{e_x} + C_{m_x} + C_{c_x} + C_{n_x} \tag{3}$$

where C_{local_x} is the total cost, the result of adding four different components:

- C_{e_x}: energy consumption costs.

$$C_{e_x} = T_{local_x} \cdot e_x \cdot p_e \cdot m \tag{4}$$

DOI: 10.1002/cpe

where e_x is the energy consumption for a given ligand x, and p_e is the energy price. Both are expressed as per unit of time.

- C_{m_x}: machine market price.

$$C_{m_x} = p/a_t \cdot T_{local_x} \cdot m \tag{5}$$

where p is the physical machine market price and a_t the amortization per unit of time. Typical values for the amortization period of a machine are 2 to 3 years. Note that, a_t is based on the unit time, that is, if the unit of time is minutes: $a_t = years \cdot 365 \cdot 24 \cdot 60$.

- C_{c_x}: Local machine facility costs.

$$C_{c_x} = \left(c_t \cdot m + A_t \cdot \left\lceil \frac{m}{m_a} \right\rceil \right) \cdot T_{local_x} \tag{6}$$

where c_t is the facility cost to support machine housing and A_t the administrator salary; both of them expressed by units of time. The adjustment is completed by taking the ceiling function value of how many physical machines are assigned to an individual administrator (m_a).

- C_{n_x}: non usage costs.

$$C_{n_x} = \left(m \cdot p/a_t + m \cdot e_i \cdot p_e + m \cdot c_t + A_t \cdot \left\lceil \frac{m}{m_a} \right\rceil \right) \cdot (1-u) \cdot T_{local_x} \tag{7}$$

where e_i is the energy consumption in idle mode and u the yearly percent machine usage rate.

4.2. Cloud model

The individual execution time per lingand (t_{cloud_x}) expressed in minutes is given by three equations shown in 8:

$$\begin{aligned} t_{cloud_A} &= 10^{-7} s_A^2 + 0.0022 s_A + 1.4518 \\ t_{cloud_B} &= 10^{-7} s_B^2 + 0.0023 s_B + 1.464 \\ t_{cloud_C} &= 10^{-7} s_C^2 + 0.0021 s_C + 1.7737 \end{aligned} \tag{8}$$

As with the local model, these formulas were obtained by fitting the results from Section 3.3.

As these times consider that a single GPU machine instance from Amazon EC2 is used, a different formula is needed for a complete infrastructure deployed in the cloud:

$$T_{cloud_x} = \frac{t_{cloud_x} \cdot l_x}{i} \tag{9}$$

where t_{cloud_x} is the time obtained for a given ligand x in Equation 8. The number of processed ligands is represented by l_x and the number of machine instances by i.

The cloud usage cost is expressed by the following formula:

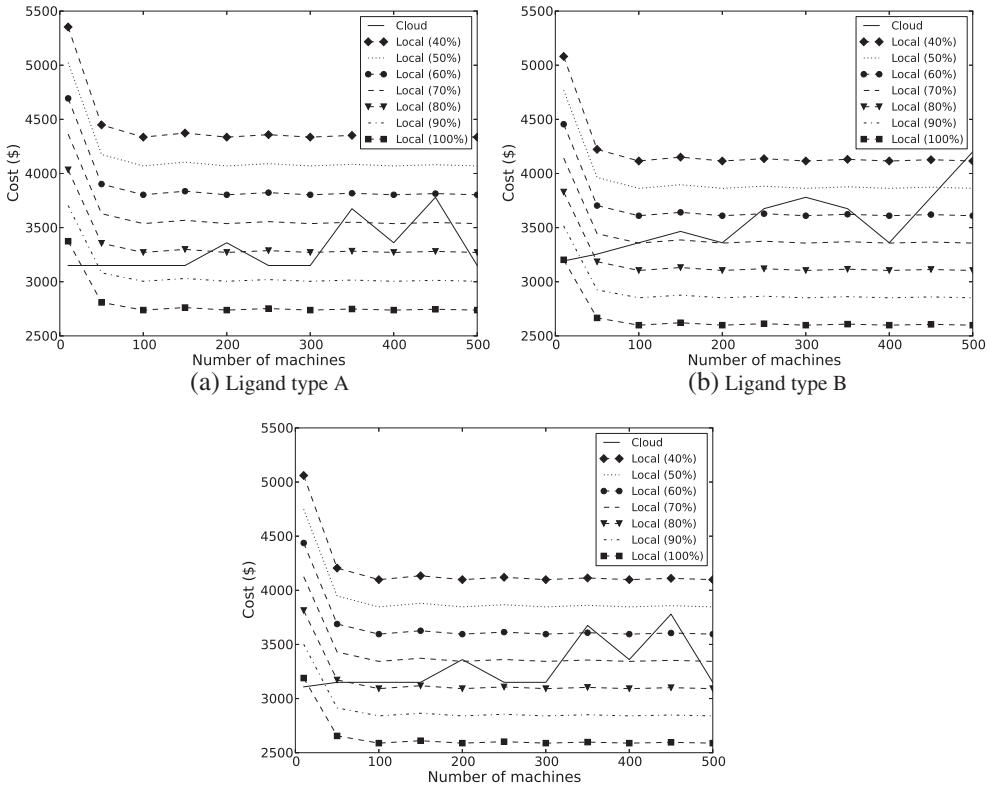
$$C_{cloud_x} = p \cdot i \cdot \left\lceil \frac{t_{cloud_x}}{60} \right\rceil \tag{10}$$

where p is the GPU cluster instance price per hour with t_{cloud_x} converted to hours. In this cloud model, a strict usage of the instantiated machines is considered; that is, they are switched off once the execution of BINDSURF is completed. However, Amazon charges per hour, for this reason the time value needs to be rounded up.

5. MODEL COMPARISON

In this Section, we compare both local and cloud models for BINDSURF processing 6,000 different ligands. Each BINDSURF simulation has 5000 Monte Carlo steps. This is the maximum number of steps we have empirically evaluated. Several assumptions are taken in order to compare those models. They are the following:

DOI: 10.1002/cpe



(a) Ligand type A (b) Ligand type B (c) Ligand type C

Figure 2. Cost values in dollars for each ligand type in the cloud infrastructure compared to different usage percentages of the local infrastructure when processing 6000 ligands and 5000 Monte Carlo steps.

- A machine from the local infrastructure costs $8,159.55.
- The amortization period of each of these machines is 3 years.
- The kW-h price is that of Spain[‖]: $0.1352.
- The energy consumption in idle mode for a machine from the local infrastructure is 245 W-h.
- The facility cost per machine per year in the local infrastructure is $12,000.
- The administrator salary is $3300/month, and each administrator is assigned to 100 machines from the local infrastructure.
- The cluster GPU instances from Amazon were launched from the US East region datacenter with a cost of $2.10/h.

Figure 2 shows the execution cost, in dollars, for the three types of ligands when increasing the number of machines. The cloud model is compared to different percentages of local infrastructure usage ranging from 40% to 100%. In the local infrastructure, the costs become stabilized from 100 machines onward. The system administrator's salary represents a rate for administering 100 machines. From this point onward, the cost is linear for the local infrastructure. Although the number of machines used in the experiments and the administrators needed to maintain those machines are increased, the execution time of the targeted application decreases.

It is noteworthy that Amazon charges per hour of use as mentioned prior ($2.10/h) and rounds up to the next whole hour. Therefore, if the execution time of an application is 1.1 h, Amazon will charge for 2 h (i.e., $4.20/h). The consequence of this rounding method is shown in Figure 2 as the price increases in accordance with the number of machines. As more machines are added to the resource pool, with the execution time equally distributed among them, it is more likely to have idle

[‖]http://www.statista.com/statistics/13020/electricity-prices-in-selected-countries/.

DOI: 10.1002/cpe

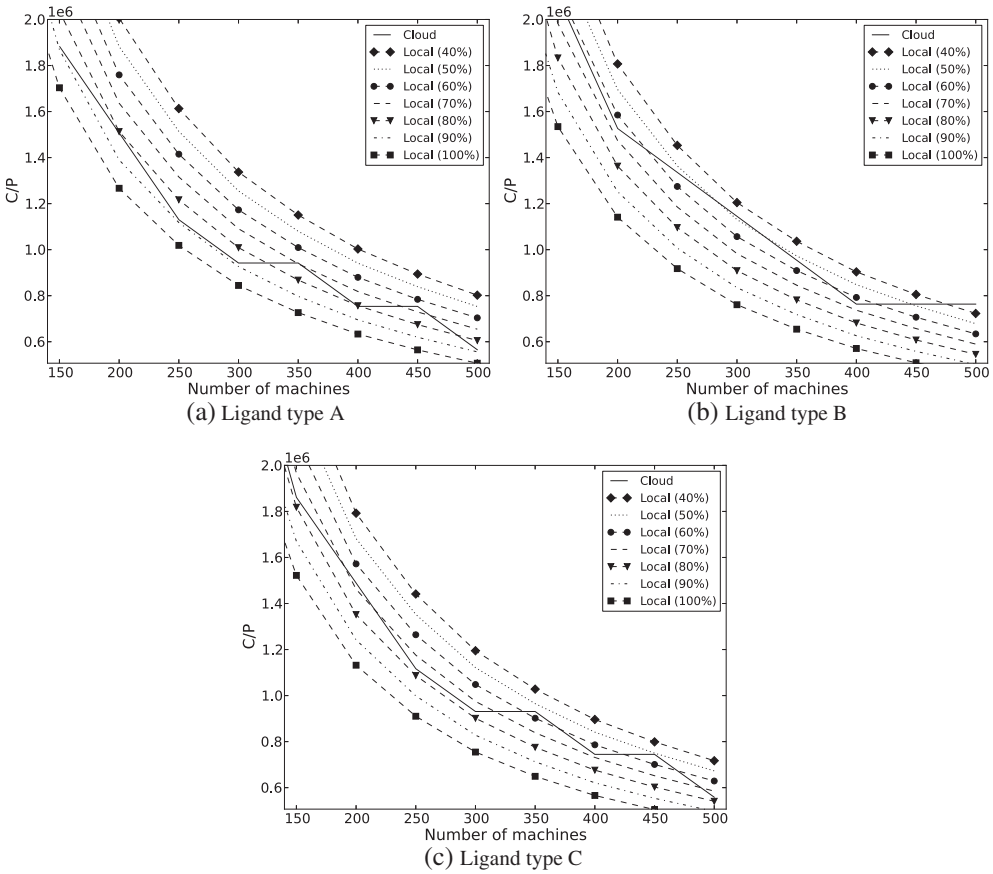


Figure 3. C/P values for each ligand type in the cloud infrastructure compared to different usage percentages of the local infrastructure when processing 6000 ligands and 5000 Monte Carlo steps.

machine hours. This fact is reflected in different behaviors of BINDSURF when executing different types of ligands. In our case, ligand type B is the most affected by the rounding method.

As the model provides performance, in terms of time, and cost values, these need to be compared. A metric, C/P [50], has been chosen for this. Its values are calculated with the following formula:

$$C/P = C \cdot T \tag{11}$$

where C and T are the cost and execution time, respectively, and is estimated by the model for a given number of machines. The best infrastructure is that with the lowest C/P value.

Figure 3 shows the C/P value for the cloud infrastructure compared to different local usage percentages ranging from 40% to 100% while increasing the number of machines. Considering an average-high usage of the local infrastructure (60%–70%), the cloud infrastructure is a good solution for ligand type A. The same happens with ligand type C but only in certain cases. The processing of ligand type B should be moved to the cloud only when an average local usage is 40% and only in very specific cases due to the rounding method used for calculating the price according to the time consumed. The graphs in Figures 2 and 3 show that system usage is the deterministic variable for system cost that drives the C/P metric.

6. CONCLUSIONS AND FUTURE WORK

We have created performance/cost models for bioinformatics using the BINDSURF algorithm in emerging research areas dependent on computational demanding tools to obtain the best execution performance for both time while optimizing costs. This work aids researchers within the field

DOI: 10.1002/cpe

of bioinformatics by helping guide which HPC platforms are best suited to run their experiments. We have evaluated two different alternatives: local infrastructure and public cloud infrastructure (Amazon) analyzing all parameters that are involved in the execution of a given application on each infrastructure.

Focusing on the physical infrastructure, we have provided a detailed cost model that considers a wide variety of elements and factors such as energy consumption, administration cost, machine facility costs, and others noted in this paper. We provided detailed comparisons of execution of the same application on the two infrastructures generating a performance/cost model for each.

The central conclusion of this work is that the machine usage per year of the local infrastructure should be quite high, ranging between 50% to 100%, in order to be profitable; otherwise, cloud computing is a more cost-effective alternative than local computing if the usage of resources is under these values. Cost calculations are different between local and cloud infrastructures as the variability in charging caused by the partial-hour upward rounding (the ceiling cost per hour) is not reflected in the local system price, and thus, cloud infrastructures are highly affected by execution time.

For future work, we plan to port BINDSURF to OpenCL [51] allowing it to be executed on a wider variety of heterogeneous computational systems such as multi-core CPUs. This will allow a wider number of, and less expensive, instance types from public cloud providers to be used for a more comprehensive performance/cost model. Additionally, other HPC-environments are emerging as a good alternative to run bioinformatic tools such as volunteer computing in which computer owners donate their computing resources to a specific project (e.g., Folding@Home).

ACKNOWLEDGEMENTS

This work has been jointly supported by the Fundación Séneca (Agencia Regional de Ciencia y Tecnología, Región de Murcia) under grant 15290/PI/2010, and by the Spanish MINECO, the European Commission FEDER funds under grants TIN2009-14475-C04 and TIN2012-31345 to G.D.G and J.M.G; the Catholic University of Murcia (UCAM) under grant PMAFI/26/12 to J.M.C. and H.P.-S.; MEDIANET (Comunidad de Madrid S2009/TIC-1468) and ServiceCloud (Ministerio de Economía y Competitividad TIN2012-31518) J.L.V.-P. Moreover, we have also used the computing facilities of Extremadura Research Center for Advanced Technologies (CETA-CIEMAT) funded by the European Regional Development Fund (ERDF), and the technical support provided by the Plataforma Andaluza de Bioinformática of the University of Málaga.

REFERENCES

1. Berman HM, Westbrook J, Feng Z, Gilliland G, Bhat TN, Weissig H, Shindyalov IN, Bourne PE. The protein data bank. *Nucleic Acids Research* 2000; **28**:235–242.
2. Sanchez R, Sali A. Large-scale protein structure modeling of the saccharomyces cerevisiae genome. *Proceedings of the National Academy of Sciences of the United States of America* 1998; **95**(23):13597–13602.
3. Garland M, Kirk DB. Understanding throughput-oriented architectures. *Communications of the ACM* 2010; **53**:58–66.
4. Garland M, Le Grand S, Nickolls J, Anderson J, Hardwick J, Morton S, Phillips E, Zhang Y, Volkov V. Parallel computing experiences with CUDA. *IEEE Micro* 2008; **28**:13–27.
5. NVIDIA. Whitepaper NVIDIA's Next Generation CUDA Compute Architecture: Kepler GK110. (Available from: http://www.nvidia.com/content/PDF/kepler/NVIDIA-Kepler-GK110-Architecture-Whitepaper.pdf) [2 August 2013].
6. The Top500 superComputers website. (Available from: http://www.top500.org/) [23 February 2013].
7. Fan X, Weber WD, Barroso LA. Power provisioning for a warehouse-sized computer. *Proceedings of the 34th Annual International Symposium on Computer Architecture*, ISCA '07, ACM: New York, NY, USA, 2007; 13–23, DOI: 10.1145/1250662.1250665.
8. Carey J. Obama's Cap-and-Trade Plan. (Available from: http://www.businessweek.com/magazine/content/09_11/b4123022554346.htm) [23 February 2013].
9. Department of Energy and Climate Change UK. CRC Energy Efficiency Scheme. (Available from: https://www.gov.uk/crc-energy-efficiency-scheme) [23 February 2013].
10. Koomey JG. Worldwide electricity used in data centers. *Environmental Research Letters* 2008; **3**(3):034008.
11. The Green500 Supercomputers Website. (Available from: http://www.green500.org/) [23 February 2013].
12. Hewitt C. ORGs for scalable, robust, privacy-friendly client cloud computing. *IEEE Internet Computing* 2008; **12**(5):96–99.

13. Berl A, Gelenbe E, Di Girolamo M, Giuliani G, De Meer H, Dang MQ, Pentikousis K. Energy-efficient cloud computing. *The Computer Journal* 2010; **53**(7):1045–1051.
14. Sánchez-Linares I, Pérez-Sánchez H, Cecilia JM, García JM. BINDSURF: a fast blind virtual screening methodology on GPUs. *Network Tools and Applications in Biology (NETTAB 2011), Clinical Bioinformatics*, Ricardo Bellazzi and Paolo Romano: Pavia (Italy), 2011; 95–97.
15. Sánchez-Linares I, Pérez-Sánchez H, Cecilia J, García J. High-throughput parallel blind virtual screening using BINDSURF. *BMC Bioinformatics* 2012; **13**(Suppl 14):S13. DOI: 10.1186/1471-2105-13-S14-S13.
16. Jorgensen W. The many roles of computation in drug discovery. *Science* 2004; **303**(5665):1813–1818.
17. Yuriev E, Agostino M, Ramsland PA. Challenges and advances in computational docking: 2009 in review. *Journal of Molecular Recognition* 2011; **24**(2):149–164.
18. Huang SY, Zou X. Advances and challenges in protein-ligand docking. *International Journal of Molecular Sciences* 2010; **11**(8):3016–3034.
19. Morris GM, Goodsell DS, Halliday RS, Huey R, Hart WE, Belew RK, Olson AJ. Automated docking using a lamarckian genetic algorithm and an empirical binding free energy function. *Journal of Computational Chemistry* 1998; **19**(14):1639–1662.
20. Friesner RA, Banks JL, Murphy RB, Halgren TA, Klicic JJ, Mainz DT, Repasky MP, Knoll EH, Shelley M, Perry JK, Shaw DE, Francis P, Shenkin PS. Glide: a new approach for rapid, accurate docking and scoring: method and assessment of docking accuracy. *Journal of Medicinal Chemistry* 2004; **47**(7):1739–1749.
21. Ewing TJA, Makino S, Skillman AG, Kuntz ID. DOCK 4.0: search strategies for automated molecular docking of flexible molecule databases. *Journal of Computer-Aided Molecular Design* 2001; **15**(5):411–428.
22. Li J, Humphrey M, Agarwal D, Jackson K, van Ingen C, Ryu Y. eScience in the cloud: a MODIS satellite data reprojection and reduction pipeline in the windows azure platform. In *2010 IEEE International Symposium on Parallel Distributed Processing (IPDPS)*. IEEE Computer Society: Los Alamitos, CA, USA, 2010; 1 –10.
23. Stevens RD, Tipney AHJ, Wroe BCJ, Oinn ATM, Senger M, Lord CPW, Goble ACA, Brass AA, Tassabehji M. Exploring Williams-Beuren syndrome using myGrid, July 2004.
24. da Cruz SMS, Batista V, Dávila AMR, Silva E, Tosta F, Vilela C, Campos MLM, Cuadrat R, Tschoeke D, Mattoso M. OrthoSearch: a scientific workflow approach to detect distant homologies on protozoans. *Proceedings of the 2008 ACM Symposium on Applied Computing*, SAC '08, ACM: New York, NY, USA, 2008; 1282–1286.
25. Krishnan A. GridBLAST: a globus-based high-throughput implementation of BLAST in a grid computing framework: research articles. *Concurrency and Computation: Practice & Experience* 2005; **17**(13):1607–1623.
26. Meyer L, Rössle S, Bisch P, Mattoso M. Parallelism in bioinformatics workflows. In *High Performance Computing for Computational Science - VECPAR 2004*, Vol. 3402, Daydé M, Dongarra J, Hernández V, Palma JM (eds), Lecture Notes in Computer Science. Springer: Berlin Heidelberg, 2005; 583–597.
27. Meyer L, Scheftner D, Voeckler J, Mattoso M, Wilde M, Foster I. An opportunistic algorithm for scheduling workflows on grids. In *High Performance Computing for Computational Science - VECPAR 2006*, Vol. 4395, Daydé M, Palma JM, Coutinho ÁL, Pacitti E, Lopes J (eds), Lecture Notes in Computer Science. Springer: Berlin Heidelberg, 2007; 1–12.
28. Andrade J, Andersen M, Berglund L, Odeberg J. Applications of grid computing in genetics and proteomics. In *Applied Parallel Computing. State of the Art in Scientific Computing*, Vol. 4699, Kågström B, Elmroth E, Dongarra J, Waśniewski J (eds), Lecture Notes in Computer Science. Springer: Berlin Heidelberg, 2007; 791–798.
29. Stockinger H, Pagni M, Cerutti L, Falquet L. Grid approach to embarrassingly parallel CPU-intensive bioinformatics problems. *Proceedings of the Second IEEE International Conference on e-Science and Grid Computing*, E-SCIENCE '06, IEEE Computer Society: Washington, DC, USA, 2006; 58.
30. Matsunaga A, Tsugawa M, Fortes J. CloudBLAST: combining mapreduce and virtualization on distributed resources for bioinformatics applications. In *Proceedings Twelfth International Conference on Intelligent Systems for Molecular Biology/Third European Conference on Computational Biology*. IEEE Computer Society: Washington, DC, USA, 2008; 222–229.
31. Lu W, Jackson J, Barga R. AzureBlast: a case study of developing science applications on the cloud. *Proceedings of the 19th ACM International Symposium on High Performance Distributed Computing*, HPDC '10, ACM: New York, NY, USA, 2010; 413–420.
32. Schatz M. BlastReduce: High Performance Short Read Mapping with MapReduce, May 2008.
33. Lee KH, Lee YJ, Choi H, Chung YD, Moon B. Parallel data processing with mapreduce: a survey. *Sigmod Record* 2012; **40**(4):11–20.
34. Dean J, Ghemawat S. MapReduce: simplified data processing on large clusters. *Communications of the ACM* 2008; **51**(1):107–113.
35. Leo S, Santoni F, Zanetti G. Biodoop: bioinformatics on hadoop. In *International Conference on Parallel Processing Workshops, 2009. ICPPW '09*. IEEE Computer Society: Washington, DC, USA, 2009; 415 –422.
36. Wang C, Zhou BB, Zomaya AY. EvolvingSpace: a data centric framework for integrating bioinformatics applications. *IEEE Transactions on Computers* 2010; **59**(6):721–734.
37. Thain D, Tannenbaum T, Livny M. Distributed computing in practice: the condor experience: research articles. *Concurrency and Computation: Practice & Experience* 2005; **17**(2–4):323–356.
38. Sul SJ, Tovchigrechko A. Parallelizing BLAST and SOM algorithms with MapReduce-MPI library. In *2011 IEEE International Symposium on Parallel and Distributed Processing Workshops and Phd Forum (IPDPSW)*. IEEE Computer Society: Los Alamitos, CA, USA, 2011; 481–489.

DOI: 10.1002/cpe

39. Qiu X, Ekanayake J, Beason S, Gunarathne T, Fox G, Barga R, Gannon D. Cloud technologies for bioinformatics applications. *Proceedings of the 2nd Workshop on Many-Task Computing on Grids and Supercomputers*, MTAGS '09, ACM: New York, NY, USA, 2009; 6:1–6:10.
40. Staten J. Hollow out the MOOSE: reducing cost with strategic rightsourcing, March 2009.
41. Gartner Research. State of Washington Agency Total Cost of IT Ownership Assessment. Garner Research, 2012. (Available from: http://ofm.wa.gov/tco/documents/final_report.pdf) [23 February 2013].
42. Kim W. Cloud Computing: Today and Tomorrow, 2009.
43. Deelman E, Singh G, Livny M, Berriman B, Good J. The cost of doing science on the cloud: the montage example. In *Proceedings of the 2008 ACM/IEEE Conference on Supercomputing*, IEEE Press, 2008; 50.
44. Hamilton J. Cooperative Expendable Micro-Slice Servers (CEMS): Low Cost, Low Power Servers for Internet-Scale Services, 2009.
45. Greenberg A, Hamilton J, Maltz DA, Patel P. The cost of a cloud: research problems in data center networks. *ACM SIGCOMM Computer Communication Review* 2008; **39**(1):68–73.
46. Kashef MM, Altmann J. A cost model for hybrid clouds. In *Economics of Grids, Clouds, Systems, and Services*. Springer: Berlin Heidelberg, 2012; 46–60.
47. Strebel J, Stage A. An economic decision model for business software application deployment on hybrid Cloud environments. In *Multikonferenz Wirtschaftsinformatik 2010*, Schumann M, Kolbe LM, Breitner MH, Frerichs A (eds). Universitätsverlag Göttingen: Göttingen, 2010; 195–206.
48. Truong HL, Dustdar S. Composable cost estimation and monitoring for computational applications in cloud computing environments. *Procedia Computer Science* 2010; **1**(1):2175–2184.
49. Navarro-Fernández J, Pérez-Sánchez H, Martínez-Martínez I, Meliciani I, Guerrero JA, Vicente V, Corral J, Wenzel W. In silico discovery of a compound with nanomolar affinity to antithrombin causing partial activation and increased heparin affinity. *Journal of Medicinal Chemistry* 2012; **55**(14):6403–6412. DOI: 10.1021/jm300621j.
50. Vazquez-Poletti J, Barderas G, Llorente I, Romero P. A model for efficient onboard actualization of an instrumental cyclogram for the Mars MetNet Mission on a public cloud infrastructure. In *Proc. Para2010: State of the Art in Scientific and Parallel Computing, Reykjavik (iceland), June 2010*, Vol. 7133, Lecture Notes in Computer Science. Springer Verlag: Berlin Heidelberg, 2012; 33–42.
51. Stone JE, Gohara D, Shi G. OpenCL: A parallel programming standard for heterogeneous computing systems. *IEEE Design & Test* 2010; **12**(3):66–73.