CONCURRENCY AND COMPUTATION: PRACTICE AND EXPERIENCE
Concurrency Computat.: Pract. Exper. 2013; **00**:1–12
Published online in Wiley InterScience (www.interscience.wiley.com). DOI: 10.1002/cpe

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, that uses the massively parallel architecture of GPUs for fast unbiased pre-screening of large ligand databases. In this contribution, we provide a performance/cost model for the execution of this application on both a physical and public cloud infrastructure. With our model it is possible to determine which is the best infrastructure by means 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.

Received ...

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 (discovery of new leads and active compounds) into meaningful health outcomes. Both objectives require to process the large data set of protein structures available in biological databases such as PDB [1] and also derived from genomic data using techniques as homology modeling [2]. Screenings in lab and compound optimization are expensive and slow methods, but bioinformatics can vastly help clinical research for the mentioned purposes by providing prediction of the toxicity of drugs and activity in non-tested targets, and by evolving discovered active compounds into drugs for the clinical trials.

This can be achieved thanks to the availability of bioinformatics tools and Virtual Screening (VS) methods that allow to test all required hypothesis before clinical trials. Nevertheless current Virtual Screening (VS) methods, such as docking, fail to make good toxicity and activity predictions being one of the main reasons that they are constrained by the access to computational resources; even the nowadays fastest VS methods cannot process large biological databases in a reasonable time-frame. Therefore, these constraints impose serious limitations in many areas of biomedical research.

The use of massively parallel hardware architectures such as Graphics Processing Units (GPUs) can tremendously overcome this problem. The GPU has become increasingly popular in the high

†These authors contributed equally.
*Correspondence to: gines.guerrero@ditec.um.es, horacio@ucam.edu

performance computing area, by combining impressive computational power with the demanding requirements of real-time graphics and 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 to their range of GPUs, with the Kepler architecture [5] being the most recent milestone in this path.

Large clusters are adopting the use of these relatively inexpensive and powerful devices as a way of accelerating computational-intensive parts of the applications. Actually, the fastest supercomputer nowadays, Titan, placed in the DOE/SC/OAK Ridge National Laboratory at Tennessee (USA) [6], is equipped with AMD Opteron Processor and the last generation of NVIDIA k20x GPUs. However, 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%, which is actually a big issue already. 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 (TCO) [7].

Moreover, reducing power consumption in these large installations is now becoming an urgent concern as several governments, like the US or the British ones, are creating taxes targeted to facilities that consume too much electricity [8, 9]. For instance, some of most famous datacenters on Internet, such as Google or 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 FLOPS (FLoating point Operations Per Second) to FLOPS per watt.

Virtualization techniques may provide significant energy savings, as they enable a larger resource usage by sharing a given hardware among several users, thus reducing the required amount of instances of that particular device. As a result, virtualization is being increasingly adopted in datacenters. In particular, cloud computing is an inherently energy-efficient virtualization technique [12], in which services run remotely in a ubiquitous computing cloud that provides 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 utilization [13]. Public cloud providers offer their services in a *pay as you go* fashion, and provide an alternative to physical infrastructures. However, this alternative only becomes real for a specific amount of data and target execution time.

In this work, we analyze this landscape of computation by targeting a GPU-based Virtual Screening method called $BINDSURF$ [14]. We propose a performance/cost model for this application that allows the user to decide which infrastructure, physical or that of a well known public cloud provider, is the 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 more computational physical resources or the more time the available resources are used, the more the total cost is increased. Moreover, an unused local infrastructure still generates costs.

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 then 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.

DOI: 10.1002/cpe

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 which ones fit an established criteria [16]. Among the many available VS methods for this purpose we decided to 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]. This 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 take the assumption, once the binding site is specified, that all ligands will interact with the protein in the same region, discarding completely the other areas of the protein.

$BINDSURF$ [15] overcomes this problem by use of GPUs by dividing the whole protein surface into arbitrary independent regions (*aka* spots). Using the parallelism of GPUs a large ligand database is screened against the target protein over its whole surface simultaneously, and docking simulations for each ligand are performed simultaneously in all the specified protein spots resulting in new spots found after the examination of 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 for the 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. The figures of cloud computing

The promise of cloud computing is delivering all the functionality of existing information technology services, and in fact it enables 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 percent of their available computing power and desktop computers at less than 5 percent 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

DOI: 10.1002/cpe

that about two-thirds of the average corporate IT staffing budget goes towards routine support and maintenance activities [41].

Cloud computing gives researchers advantages from the convergence of computational resource efficiency where the power of modern computers is 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, keeping service level agreements improves large scale computing acceptability with greater environmental sustainability [43]. In the case of Amazon, estimates by Hamilton from Amazon Services [44], the cost and operation based on a three-year amortization schedule (the low end of the industry nominal schedule of replacement every 3-7 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.

parts of the cloud and the aggregation of a cloud resources can provide higher hardware usage.

3. EXPERIMENTS

3.1. Experiment definition

We carried out VS calculations using $BINDSURF$ for the direct prediction of binding poses. For this purpose we chose 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 co-factor 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 50000. 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 since sometimes we might be interested in short simulations ($steps = 5, 10, 50$) for obtaining qualitative information about potential hot-spots in the surface screening approach for millions of different ligands, but in some other situations we might be more interested in obtaining accurate predictions for a smaller set of ligands, and then use higher values for the $steps$ parameter such as $500, 5000$ and 50000. The outcome of a docking simulation performed by $BINDSURF$ for a type A ligand (PDB ID is $1qcf$) is shown in Figure 1.

3.2. Infrastructure used

Table I. Platforms System Specifications.

(a) Local Machine

Proc.:	Intel Xeon E5620@2.4Ghz
Memory:	16GB
2xGPU NVIDIA Tesla C2050	
GPU:	GF100
Memory Size:	3072 MB
Memory Bandwidth:	144 GB/sec
Stream Processors:	448
Max Power Draw:	238 W

(b) Amazon EC2

Proc.:	2xIntel Xeon X5570@2.93GHz
Memory:	22GB
2xGPU NVIDIA Tesla M2050	
GPU:	GF100
Memory Size:	3072 MB
Memory Bandwidth:	148.4 GB/sec
Stream Processors:	448
Max Power Draw:	225 W

DOI: 10.1002/cpe

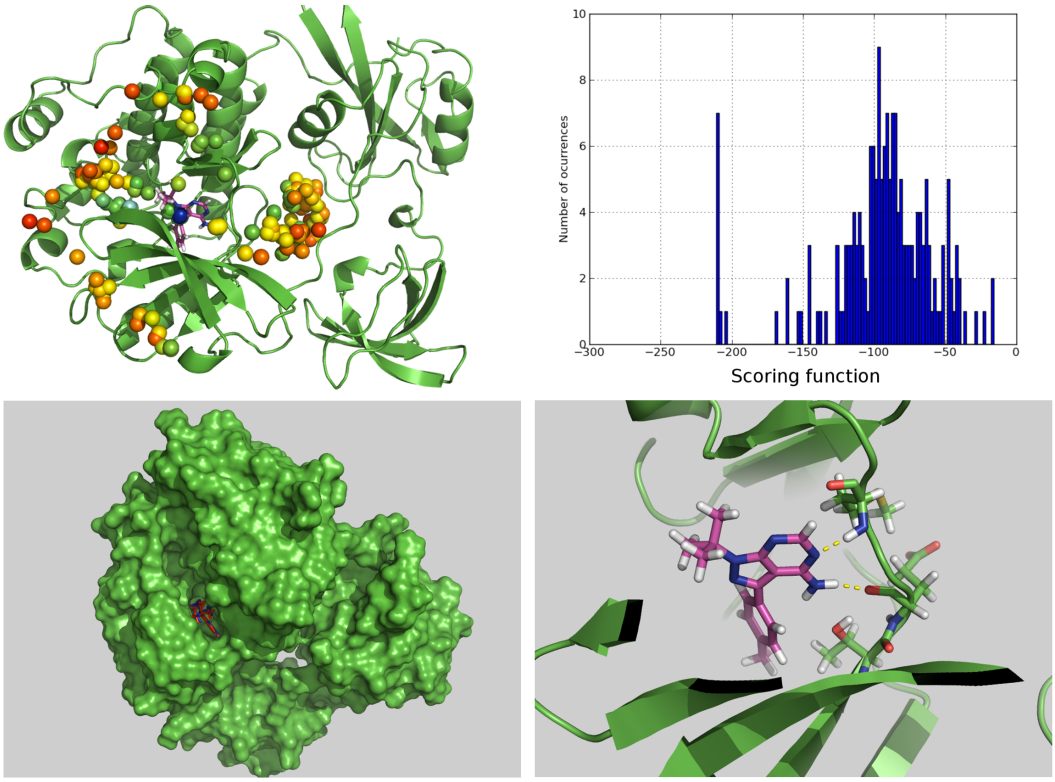


Figure 1. Surface screening results for PDB:1QCF. From up left to down right; a) beads represent protein spots and the color of each bead is related with the value of the scoring function, so colors from red to blue indicate lower values for the scoring function, b) histogram with the distribution of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

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

The chosen cloud infrastructure is the one offered by Amazon through its Elastic Compute Cloud services[†]. As BINDUSRF is coded using CUDA, the Cluster GPU instances were the only possible choice. The specifications of the GPU provided by Amazon EC2 are displayed on 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 per hour[‡].

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 the experiments we have performed on both local and cloud infrastructure before we develop the local and cloud model. These experiments are based on ten 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. Execution times on the cloud are higher than in the local infrastructure, yielding ligand type B (2byr) the lowest values and ligand type C

[†]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

(3p4w) the highest. As expected, local infrastructure defeats by tight margin cloud infrastructure mainly due to the overhead introduced by virtualization and communications in the cloud.

Table III. Power consumption for one ligand of each type with different number of simulation steps in the 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

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

4. EXECUTION MODEL

This Section shows the performance/cost model defined using the results obtained in the previous Section. The model is categorized by infrastructure (local, cloud) and then by ligand type (l1c4, 2byr, 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 at the very beginning.

4.1. Local Model

The individual execution time, expressed in minutes, for each ligand of the local model is given by the formulas 1. These formulas were obtained by fitting the results from Section 3.3.

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

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

Given a number of physical machines m and the processed ligands of a particular type, the Equation 2 shows the extrapolated total execution time.

$$T_{local_x} = \frac{t_{local_x}\cdot l_x}{m} \quad (2)$$

DOI: 10.1002/cpe

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.

The local costs can be expressed by the following formula:

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

being the total cost the result of adding four different components:

- C_{e_x}: energy consumption costs.

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

  being e_x the energy consumption for a given ligand x and p_e the energy price. Both are expressed per unit of time.
- C_{m_x}: machine market price.

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

  being p the physical machine market price and a_t the amortization per unit time. Typical values for the amortization period of a machine are 2-3 years. Note that, a_t is based on the unit time, i.e. if the unit time is minutes $a_t = years \cdot 365 \cdot 24 \cdot 60$.
- C_{c_x}: machine collocation costs.

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

  being c_t the collocation and A_t the administrator salary, both of them expressed by units of time. The adjustment is completed by specifying how many physical machines are assigned to an individual administrator (m_a). The expression $\lceil x \rceil$ corresponds to the ceiling function of x.
- C_{n_x}: non usage costs.

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

  being e_i the energy consumption in idle mode and u the yearly usage rate.

4.2. Cloud Model

The individual execution time per lingand (t_{cloud_x}) expressed in minutes is given by the following formulas:

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

Like in the local model, these formulas were obtained by fitting the results from Section 3.3.

As these times consider that only a GPU machine instance from Amazon EC2 is used, another 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.

On the other hand, the cloud usage cost can be expressed by the following formula:

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

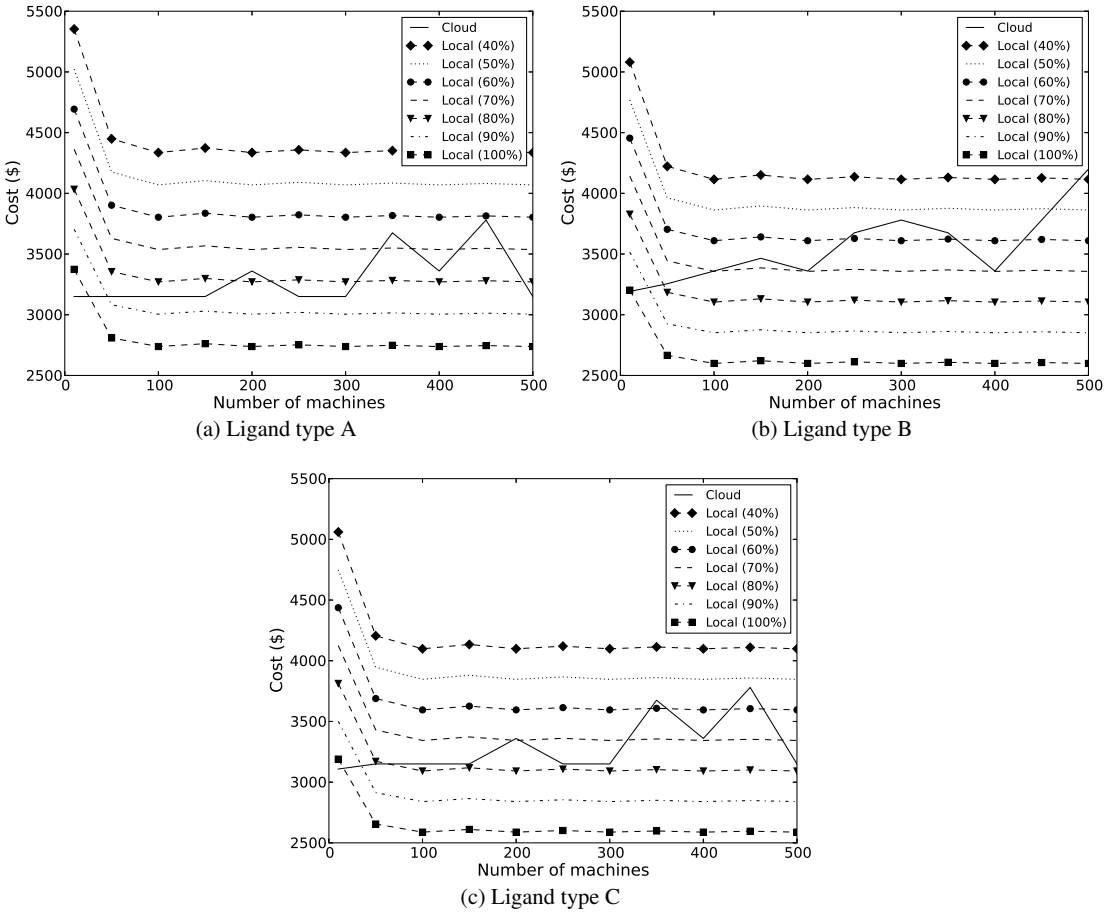
where p is the GPU cluster instance price per hour. Also, t_{cloud_x} needs to be converted to hours. In this cloud model a strict usage of the instantiated machines is considered, that is, they are switched off once the execution of $BINDSURF$ is completed. However, Amazon charges per hour, for this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

DOI: 10.1002/cpe

5. MODEL COMPARISON

In this Section, we compare both the local and the cloud model by executing BINDSURF for processing 6,000 different ligands. The number of Montecarlo steps for each BINDSURF simulation is 5,000. This is the maximum number of steps we have empirically evaluated. Several assumptions are taken in order to compare those models. They are:

1. A machine from the local infrastructure costs $8,159.55.
2. The amortization period of each of these machines is 3 years.
3. The KW/h price is that of Spain[§]: $0.1352.
4. The energy consumption in idle mode for a machine from the local infrastructure is 245W/h.
5. The collocation price per machine/year in the local infrastructure is $12,000.
6. The administrator salary is $3,300/month and each administrator is assigned to 100 machines from the local infrastructure.
7. The Cluster GPU instances from Amazon were launched at the US East region with a cost of $2.10/h.



(a) Ligand type A

(b) Ligand type B

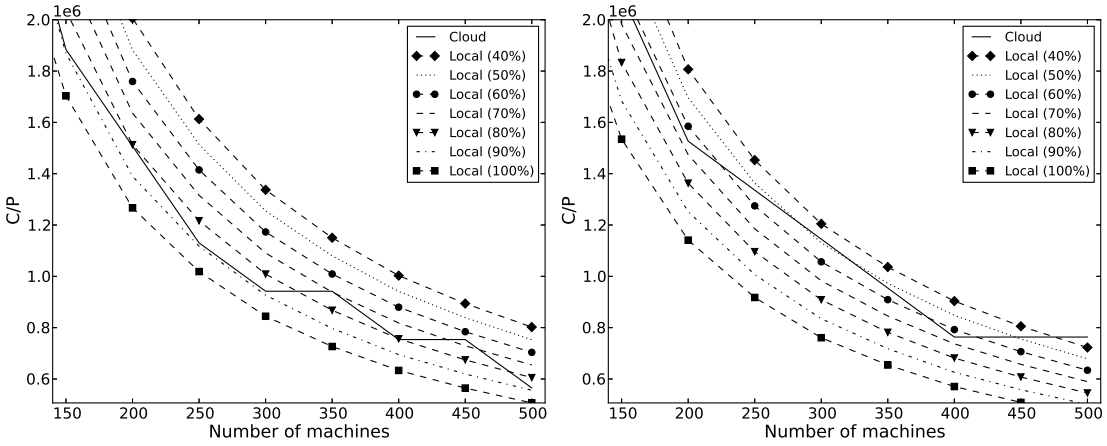
(c) Ligand type C

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 6,000 ligands and 5,000 Monte Carlo steps.

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 percentage of the local infrastructure

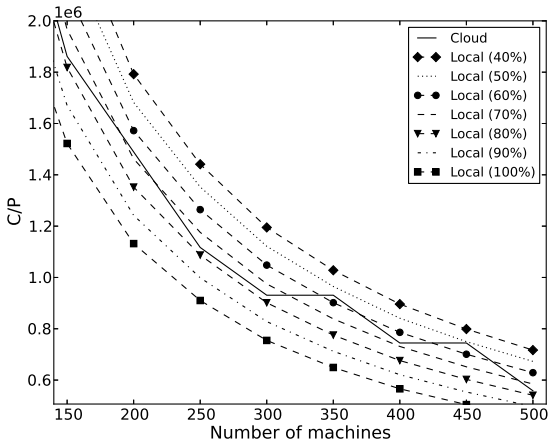
[§]http://www.statista.com/statistics/13020/electricity-prices-in-selected-countries/

DOI: 10.1002/cpe



(a) Ligand type A

(b) Ligand type B



(c) Ligand type C

Figure 3. C/P values for each ligand type in the cloud infrastructure compared to different usage percentages of the local infrastructure when processing 6,000 ligands and 5,000 Monte Carlo steps.

usage ranging from 40% to 100%. In the local infrastructure, the costs become stabilized from 100 machines onwards. The administration salary covers his minimum wage that represents to deal with 100 machines. From this point onwards, the cost is linear in the local infrastructure. Although the number of machines used in the experiments and administrators needed to maintain those machines are increased, the execution time of the targeted application decreases as well.

It is noteworthy that Amazon charges per hour of use as previously mentioned ($ 2.1per hour). Therefore, if the execution time of your application is 1.1 hour, Amazon will charge you 2 hours (i.e. $ 4.2 per hour). The consequence of this rounding method is shown in Figure 2 as price increases in accordance with the number of machines. As long as more machines are added to the resource pool, the execution time is equally distributed among them, and thus, it is likely to have more idle hours. This fact is also reflected in different behaviors of $BINDSURF$ when executing different types of ligands; i.e. types A, B and C. In this case, ligand type B is the most affected by the rounding method.

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

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

DOI: 10.1002/cpe

where C and T are the cost and execution time 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 that of 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%), cloud turns to be a good solution for ligand type A. The same happens with ligand type C but in certain cases. The process of ligand type B should be moved to the cloud only with an average local usage (40%) and in very specific cases. This is mainly due to the rounding method used for calculating the price according to the time consumed. The graphs show that system usage is the deterministic variable for system cost that drives the C/P metric.

CONCLUSIONS AND FUTURE WORK

Bioinformatics is an emerging research area which generates many computational demanding tools. Obtaining the most performance in their execution is important but optimization through cost reduction is crucial, even more so with the current economic context. This work aids researches within the field of Bioinformatics to decide which HPC platforms is the best suited to run their experiments on. We evaluate two different alternatives: local infrastructure and public clouds providers like Amazon, analyzing all parameters that are involved in the execution of a given application on each of them.

Focusing on the physical infrastructure, we have provided an exhaustive cost model that considers a wide variety of elements and factors, such as energy consumption, administration cost, machine collocation cost, etc. This allows us to provide a detailed comparison with the execution of the same application on the infrastructure provided by Amazon, generating a performance/cost model for them. The main conclusion of this work is that the resource occupancy per year of the local infrastructure should be quite high (ranging between 50% to 100%) in order to be profitable. Otherwise, cloud computing is most-cost effective alternative than local computing if the utilization of resources is under these values. Nonetheless, Cost calculations are different between local and cloud systems as the variability in charging (the ceiling cost per hour) is not reflected in the local system price, and thus, it is drastically affected by execution time.

As future work, we plan to port $BINDSURF$ to OpenCL [51], allowing its execution to a wider variety of heterogeneous computational systems such as multicore CPUs. This way, more and cheaper instance types from public cloud providers could be harnessed and a more complete performance/cost model could be fostered. Moreover, 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.

ACKNOWLEDGMENTS

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 Centre 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

DOI: 10.1002/cpe

1. Berman HM, Westbrook J, Feng Z, Gilliland G, Bhat TN, Weissig H, Shindyalov IN, Bourne PE. The Protein Data Bank. *Nucleic Acids Res* 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* Nov 1998; **95**(23):13 597–13 602.
3. Garland M, Kirk DB. Understanding Throughput-Oriented Architectures. *Commun. ACM* November 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* July 2008; **28**:13–27.
5. NVIDIA. *Whitepaper NVIDIA's Next Generation CUDA Compute Architecture: Kepler GK110*. NVIDIA, 2012.
6. The Top500 superComputers website. `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. John Carey. Obama's Cap-and-Trade Plan. `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. `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):034 008.
11. The Green500 superComputers website. `http://www.green500.org/` [23 February 2013].
12. Hewitt C. ORGs for Scalable, Robust, Privacy-Friendly Client Cloud Computing. *IEEE Internet Computing* Sep 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 and Goodsell, DS and Halliday, RS and Huey, R and Hart, WE and Belew, RK and 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, *et al.*. 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. *Parallel Distributed Processing (IPDPS), 2010 IEEE International Symposium on*, 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 Jul 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. *Concurr. Comput. : Pract. Exper.* Nov 2005; **17**(13):1607–1623.
26. Meyer L, Rˆssle S, Bisch P, Mattoso M. Parallelism in Bioinformatics Workflows. *High Performance Computing for Computational Science - VECPAR 2004*, *Lecture Notes in Computer Science*, vol. 3402, Daydé M, Dongarra J, Hernández V, Palma JM (eds.). 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. *High Performance Computing for Computational Science - VECPAR 2006*, *Lecture Notes in Computer Science*, vol. 4395, Daydé M, Palma JM, Coutinho ÁL, Pacitti E, Lopes J (eds.). Springer Berlin Heidelberg, 2007; 1–12.
28. Andrade J, Andersen M, Berglund L, Odeberg J. Applications of Grid Computing in Genetics and Proteomics. *Applied Parallel Computing. State of the Art in Scientific Computing*, *Lecture Notes in Computer Science*, vol. 4699, Kågström B, Elmroth E, Dongarra J, Waśniewski J (eds.). 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. *Proceedings Twelfth International Conference on Intelligent Systems for Molecular Biology/Third European Conference on Computational Biology*, 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 Rec.* Jan 2012; **40**(4):11–20.
34. Dean J, Ghemawat S. MapReduce: Simplified Data Processing on Large Clusters. *Commun. ACM* Jan 2008; **51**(1):107–113.
35. Leo S, Santoni F, Zanetti G. Biodoop: Bioinformatics on Hadoop. *Parallel Processing Workshops, 2009. ICPPW '09. International Conference on*, 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. *Concurr. Comput. : Pract. Exper.* Feb 2005; **17**(2-4):323–356.
38. Sul SJ, Tovchigrechko A. Parallelizing BLAST and SOM Algorithms with MapReduce-MPI Library. *Parallel and Distributed Processing Workshops and Phd Forum (IPDPSW), 2011 IEEE International Symposium on*, 2011; 481 –489.
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 Mar 2009.
41. Gartner Research. State of Washington Agency Total Cost of IT Ownership Assessment Jul 2012. `http://ofm.wa.gov/tco/documents/final\_report.pdf` [23 February 2013].
42. Kim W. Cloud Computing: Today and Tomorrow Jan 2009.
43. Deelman E, Singh G, Livny M, Berriman B, Good J. The cost of doing science on the cloud: the montage example. *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 Jan 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. *Economics of Grids, Clouds, Systems, and Services*. Springer, 2012; 46–60.
47. Strebel J, Stage A. An economic decision model for business software application deployment on hybrid cloud environments. *Multikonferenz Wirtschaftsinformatik 2010* 2010; :47.
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. *Proc. PARA2010: State of the Art in Scientific and Parallel Computing, Reykjavik (Iceland), June 2010*, *Lecture Notes in Computer Science*, vol. 7133, Springer Verlag, 2012; 33–42.
51. Stone JE, Gohara D, Shi G. OpenCL: A Parallel Programming Standard for Heterogeneous Computing Systems. *IEEE Des. Test* May 2010; **12**(3):66–73.