Journal of Structural Biology 200 (2017) 20–27

Contents lists available at ScienceDirect

Journal of Structural Biology

journal homepage: www.elsevier.com/locate/yjsbi

ScipionCloud: An integrative and interactive gateway for large scale cryo electron microscopy image processing on commercial and academic clouds

Jesús Cuenca-Alba [a,*], Laura del Cano [a,1], Josué Gómez Blanco [a], José Miguel de la Rosa Trevín [a], Pablo Conesa Mingo [a], Roberto Marabini [b], Carlos Oscar S. Sorzano [a], Jose María Carazo [a]

[a] Centro Nacional de Biotecnología (CNB-CSIC), Cantoblanco, 28049 Madrid, Spain
[b] Escuela Politécnica Superior, Universidad Autónoma de Madrid, Cantoblanco, 28049 Madrid, Spain

ARTICLE INFO

Article history:
Received 11 November 2016
Received in revised form 23 May 2017
Accepted 12 June 2017
Available online 26 June 2017

Keywords:
Cryo-electron microscopy
Cloud computing
Distributed computing

ABSTRACT

New instrumentation for cryo electron microscopy (cryoEM) has significantly increased data collection rate as well as data quality, creating bottlenecks at the image processing level. Current image processing model of moving the acquired images from the data source (electron microscope) to desktops or local clusters for processing is encountering many practical limitations. However, computing may also take place in distributed and decentralized environments. In this way, cloud is a new form of accessing computing and storage resources on demand. Here, we evaluate on how this new computational paradigm can be effectively used by extending our current integrative framework for image processing, creating ScipionCloud. This new development has resulted in a full installation of Scipion both in public and private clouds, accessible as public "images", with all the required preinstalled cryoEM software, just requiring a Web browser to access all Graphical User Interfaces. We have profiled the performance of different configurations on Amazon Web Services and the European Federated Cloud, always on architectures incorporating GPU's, and compared them with a local facility. We have also analyzed the economical convenience of different scenarios, so cryoEM scientists have a clearer picture of the setup that is best suited for their needs and budgets.

1. Introduction

We are witnessing a revolution in the field of 3D Electron Microscopy under cryogenic conditions (cryoEM), mainly due to the introduction of direct electron detectors together with continuous improvements in computing hardware and in software tools (Vinothkumar and Henderson, 2016). A single modern electron microscope produces around 1 TB of data per day that currently requires at the very least 1000 CPU-hours of image processing (Scheres, 2014). As a consequence, scientists are investigating how different technologies can support the next generation of image processing software. These technologies include processing technologies such as graphical processing units (GPUs), and platforms such as clouds. In this work we concentrate on the use of cloud computing in configurations incorporating GPU's.

There are two main types of cloud providers: commercial and academic. In this work we will develop and test our implementation on Amazon Web services (AWS) (one of the best known commercial providers) and on the European Federated Cloud (FedCloud) (Fernández-del Castillo et al., 2015) which is, at least partially, free of charge for academic users (in Europe).

AWS provides a broad set of services, among which we are mainly interested on "Infrastructure as a Service" (IaaS) services, such as computing, storage and networking. They are offered as an utility: on-demand, available within seconds and with pay-as-you-go pricing. Very few reports in the cryoEM field deal with cloud implementations (e.g. Cianfrocco and Leschziner, 2015), but from these few (including this one), AWS is becoming the platform of choice in Structural Biology for High-performance computing (HPC) on the cloud.

FedCloud, in turn, is an IaaS cloud provider for research, backed by the European Grid Initiative (EGI, https://www.egi.eu). This research e-infrastructure is organized as a set of Virtual Organizations, which typically represent large scale research projects. Resources are also available on demand, but the researcher is not

* Corresponding author.
E-mail address: jcuenca@cnb.csic.es (J. Cuenca-Alba).
[1] Co-first author.

https://doi.org/10.1016/j.jsb.2017.06.004
1047-8477/

always billed. EGI acts as a mediator between Virtual Organizations and resource providers, negotiating billing procedures and prices of the resources, which can then be offered for free, or with a pay-as-you-go model. Acknowledgment of these resources is expected in published research papers.

Although cloud computing and GPU implementations may satisfy the growing needs of processing power of cryoEM, a clear layer of project and data management is also required, as it is a system able to bridge and integrate among disparate image processing packages. Traceability and reproducibility of all image processing steps to get a final 3D map is also a must. This is the context in which Scipion has been developed (de la Rosa-Trevín, 2016).

Scipion is an image processing framework aimed at obtaining 3D maps of macromolecular complexes using cryo Electron Microscopy. It has emerged as the solution offered by the Instruct Image Processing Center (I2PC) to European scientists accessing the European Research Infrastructure for Structural Biology (Instruct). With Scipion, researchers can obtain 3D maps of macromolecular complexes combining the best from most popular cryoEM software packages. All low-level details (such as taking care of formats and conversions, or tracking the parameters of each step) are automatically handled. The Graphical User Interface (GUI) of Scipion provides homogeneous access to all the underlying packages, enhanced with rich data viewers, wizards and other practical tools. Scipion encourages reproducibility since every step in the processing pipeline is logged, creating a workflow that may be stored or shared with other researchers. Scipion encapsulates distributed computing details, so running workflows in clusters of computing nodes is quite straightforward.

In this work we present "ScipionCloud", a gateway for running Scipion on cloud. Some of the benefits of ScipionCloud are:

- End-users only need a web browser on their local computer to use ScipionCloud. Installation, configuration, resource management, etc are all transparently handled.
- Complete traceability of results is provided, from the initial images to the final 3D map.
- Computational and graphical performance on cloud is similar to equivalent local infrastructures, overcoming traditional performance issues regarding HPC on cloud.
- ScipionCloud adapts to single node and cluster configurations in the cloud, making the most of the resources available (including hardware accelerators, like GPUs).
- Projects can easily be migrated across cloud providers and local resources. Researchers can perform the more interactive steps on their local computer and then continue processing on the cloud, or vice versa.
- ScipionCloud easily accommodates to the current trend of concentrating computational resources in big facilities (whether commercial, as AWS, or academic, as FedCloud).

2. Results

2.1. ScipionCloud

Scipion can be installed from precompiled binaries or built from source code. It offers an automatic mechanism to install the EM packages that are integrated in the framework. Until now the approach described above has been the usual way to install Scipion, both in personal computers and in large clusters, but the arrival of cloud computing has offered new possibilities for software distribution, and it is in this context that ScipionCloud has been developed.

ScipionCloud is available as an Amazon Machine Image (AMI) -that can be used on AWS platform-, and as a EGI Virtual Appliance. In both Amazon Machine Image (AMI) and Virtual Appliance cases, we offer a full Scipion installation plus a number of useful utilities to facilitate users experience. Supplementary Table 1 shows the list of packages and libraries included in the first version of ScipionCloud.

ScipionCloud can be easily deployed following the documentation at the Scipion project wiki (https://github.com/I2PC/scipion/wiki/ScipionCloud#how-to-use-scipioncloud). Once a user has available a set of cloud resources, the procedure to use ScipionCloud is, essentially:

1. Launch a ScipionCloud instance.
   (a) For Amazon AWS, we recommend to use the AWS console accessed through a web interface. The basic steps are: (a) Choose the ScipionCloud AMI in "Community AMIs" section, (b) choose the instance type, according to how much resources (CPU, RAM, GPU and disk) are needed, (c) add extra storage (check our recommendations along this work), (d) configure security (allow inbound connections via SSH and HTTPS) and, finally, (e) launch the instance. If the user intends to set up a HPC cluster, this has to be deployed using Starcluster, as described in the ScipionCloud documentation.
   (b) For EGI FedCloud, we recommend to use the OCCI client to create the instances (https://wiki.egi.eu/wiki/HOWTO11), either as a standalone program, or from an OCCI virtual appliance.
2. Once the instance is running, the steps are the same, independently of where ScipionCloud is actually executing: (a) Connect to the instance with ssh to grab the randomly-generated remote display password and (b) use this password to authenticate in the browser.
3. Upload your data through ssh or rsync (more sophisticated data transfer procedures are also possible, check on our recommendations).
4. Start Scipion.

2.2. ScipionCloud profiling and cost analysis

We have taken the position of a typical user that wants to know in detail how efficient (and how expensive) is the use of cloud computing -ScipionCloud in this case- in common image processing scenarios. To this end we have analyzed the performance and cost of running a typical Single Particle Image processing workflow on both public and private clouds as well as on a local computer, always with the constraint of achieving comparable results in term of quasi atomic resolution. The comparative of the different ScipionCloud profilings will be presented in detail in the following.

Fig. 1 shows the precise workflow used for this test. Essentially, 966 videos corresponding to samples of mammalian transient receptor potential ion channel (TRPV1) were subjected to a process of (a) Frame alignment (correction of Beam Induced Movement in videos), (b) CTF estimation, (c) Interactive and automatic particle picking (these interactive steps were tested but existing coordinates were imported to continue with the workflow), (d) Relion 2D classification, (e) Relion 3D classification and (f) Relion 3D auto-refine to produce a 3D map of 3.4 Å. Videos correspond to the study performed by Liao et al. (2013), and deposited in EMPIAR with entry number 10005. We include the complete workflow actually used in Supplemental Materials as a *json* file produced with the 'Export Workflow' functionality of Scipion.

We have run that same EM workflow on a commodity server to provide the user with a clear anchor point to well-known computational environments, complementing the following Cloud scenarios:

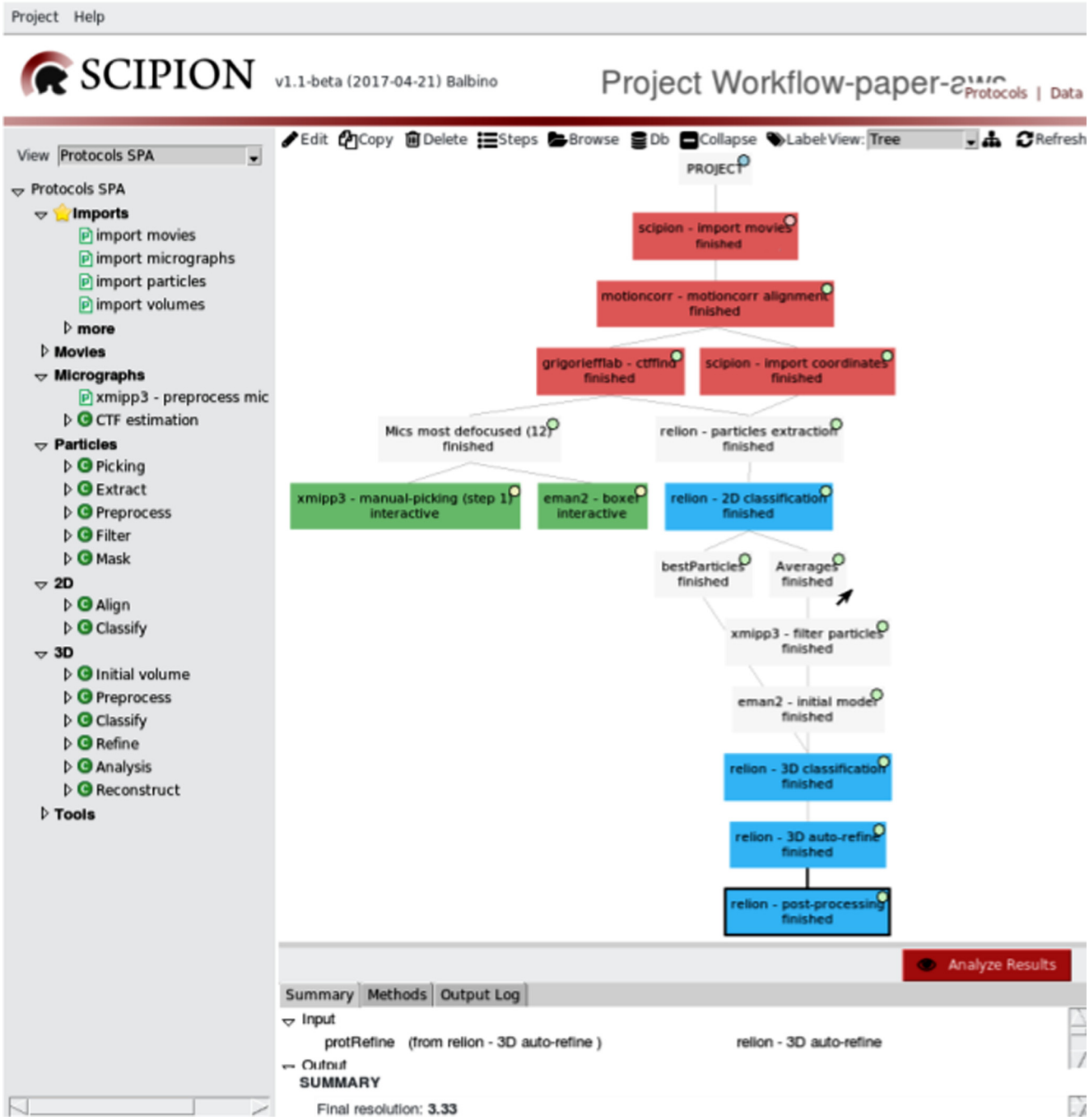

Fig. 1. Single Particle Analysis image processing workflow used in this work. The different steps are shown within boxes. Shown in red the steps labelled as Preprocessing (all performed in streaming) and in blue the most important steps of the Processing. Note that Scipion streaming mode allows the overlapping of communication and computation, which is specially interesting in enviroments such as Cloud.

- AWS single instance with GPUs
- EGI FedCloud single instance with GPUs

Table 1 presents the description of the instance types used for the benchmark. Note that in the case of Amazon AWS, the choice of instances cover the demands of most potential users. Indeed, AWS instances range from *t2.nano* (1 vCPU and 0.5 GB RAM) to *x1.32xlarge* (128 vCPU and 1952 GB RAM), and from *g2.2xlarge* (1 GPU NVIDIA Grid K520 4 GB) to *p2.16xlarge* (16 GPU NVIDIA Tesla K80 12 GB). While *g2.2xlarge* may look huge to the *t2.nano* user, it is actually "small" for many EM processing scenarios.

The particularities of some of the workflow steps have required the use of different instance types, but it should be noted that the Amazon AWS EC2 console allows to easily change the type of an existing virtual machine. On the contrary, the Federated Cloud does not currently provide a simple mechanism to do so, but the use of external Block Storage to actually store projects and data can simplify switching from one instance machine to a new one if a change of type is beneficial.

Common to all cloud environments is the need to first upload the initial Electron Microscopy images to be used. We performed five network tests, which involved CNB-CSIC in Madrid, the

Table 1
Instance types summary. Amazon cost reflects on-demand instance prices at AWS EU Ireland region as May 2017. The cost of titanxp is an estimation that includes the price of the server, electricity, housing and personnel (see "Discussion"). vCPU stands for Virtual CPUs.

Environment	Instance	CPU model	vCPU	Cores	GPU	RAM(GB)	Cost ($/h)
AWS EC2 Ireland	g2.2xlarge	E5-2670v2	8	4	K520 4 GB	15	0.702
	p2.8xlarge	E5-2686v4	32	16	K80 12 GB	488	7.776
FedCloud IISAS	gpu1cpu6	E5-2650v2	6	6	K20 4 GB	24	–
	gpu2cpu12	E5-2650v2	12	12	K20 4 GB	48	–
Local	titanxp	ES-2630v3	32	24	Titan XP 12 GB	128	1.4

European Bioinformatics Institute in Cambridge, the FedCloud datacenter in Czech Republic, and the Amazon datacenter in Ireland. The bandwith of the CNB-CSIC data connection is 1 Gb/s (or 125 MB/s). We started using a well-known and public domain protocol combination of rsync over ssh, but the performance was not as good as in the case of employing optimized data transfer tools, like bbcp -public domain- (www.slac.stanford.edu/∼abh/bbcp) or Aspera FASP -commercial- (http://asperasoft.com/technology/transport/fasp). The average speed with both optimized tools was similar, but FASP showed higher peaks (up to 110 MB/s, almost saturating CNB-CSIC connection). Considering that Aspera FASP is a relatively expensive software solution, that bbcp is public domain, and that their performance in our specific case was similar, we recommend to use bbcp. However, in this work, we have started the processing transferring the microscope movies from the EMPIAR repository (http://www.ebi.ac.uk/pdbe/emdb/empiar/), which allow to use Aspera connect client at no cost. With such speeds, it takes approximately 80 s to upload a single 6 GB movie, and 21 h to upload the whole movie set. In terms of cost, the uploading, per se, was free of charge, but there were some relatively small computing and storage costs that were charged in AWS.

We present the results obtained regarding data transfer rates on Table 2. We follow the convention of representing bits with a lowercase b and bytes with an uppercase B. Thus, MB/s stands for Megabytes per second, and Gb/s for Gigabits per second.

Before presenting the results of the computational profiling for each platform, we note the following general considerations:

- The number of CPUs and GPUs used for each step was chosen considering several factors, like memory requirements and the range of configurations offered by each cloud provider, always with the aim of minimizing economic costs and execution time.
- Some of the steps could have been executed on a less powerful type (specially regarding the use of a GPU) but, considering the short processing times, the tiny savings in cost did not compensate the time and effort of switching to another type, even if facilitated by the AWS console and the use of external storage. For example, Ctffind4 did not need a GPU and Relion postprocessing could have been run on a smaller type.
- On both cloud scenarios processing started by transferring the movies, which, even using Aspera, took a long time. In order not to waste this time, the "streaming mode" offered by Scipion was used, which allowed to perform the next two steps (Frame alignment and Ctf estimation) while the movies were being transferred.
- EM projects demand far more storage than the 30 GB available in the local disk of the ScipionCloud default instance. Therefore, we have always attached external volumes to store the raw data as well as the Scipion project. On AWS A 10 TB volume was attached only during preprocessing, since the complete videos had to be transferred and stored, and then project was moved to a 1 TB disk to reduce costs. On the FedCloud we could not have a disk bigger than 2 TB, so we removed movies after they were being processed on Streaming mode, thus liberating some space. We note that we tested the performance of local disks compared to attached volumes, and that there was a degradation of about 20 per cent in those steps of the workflow involving intense disk access.
- In all cases interactivity was excellent, even in the demanding task of particle picking, thanks to the use of an intermediate software layer encapsulating X11 graphics, as it is further presented under Material and Methods.

Thanks to the Scipion project-oriented features, we could run exactly the same complete EM worflow on all GPU-based configurations. Naturally, results, measured as finally achieved resolution, were always the same for all local and Cloud configurations.

First, we used a single node with GPU on AWS. We started with a *g2.2xlarge* instance (1 GPU NVIDIA GRID K520) to execute the first part of the workflow (Acquisition and Preprocessing). Then, we switched to a more powerful (and expensive) instance, *p2.8xlarge* (8 GPU NVIDIA Tesla K80) to process the particles and obtain the final map.

Second, we used Federated Cloud *gpu1cpu6* and *gpu2cpu12* machine types.

Finally, we used a *titanxp* local commodity server (2 GPU NVIDIA Titan XP).

Results corresponding to the GPU profiles are presented in Table 3.

Fig. 2 summarizes the results showing the total elapsed time (normalized by MPI) of the processing part of the workflow for each profile as well as the distribution of on-demand costs.

3. Discussion

After having done profiling tests of ScipionCloud on several cloud architectures as well on local ones, several questions came to our minds, such as: (1) How does local and cloud computing compare for similar architectures? (2) How long will it take to

Table 2
Data transfer rates for image upload.

From	To	Protocol	Avg Speed (MB/s)	Max Speed (MB/s)
CNB (Spain)	AWS Ireland	rsync+ssh	30	30
		bbcp	75	87
EBI (UK)	AWS Ireland	Aspera FASP	75	110
AWS Ireland	AWS Ireland	rsync+ssh	30	30
CNB (Spain)	CESNET FedCloud	rsync+ssh	37	37

Table 3
Results for ScipionCloud GPU profiling. Note that for Amazon we report on-demand pricing; consumed partial instance-hours are billed as full hours. Thr. stands for threads, ELT stands for Elapsed time, and Cost inst. for on-demand Cost instance.

Step	Node type	GPUs	Thr.	MPI	Disk (TB)	ELT (h)	Cost ins. ($)	Cost disk($)
AWS instance GPU								
Movie transfer (Aspera)	g2.2xlarge	–	–	–	10	36.7	103	55.53
Movie align (Motioncor2)	g2.2xlarge	1	–	–	10			
Ctf estimation (ctffind4)	g2.2xlarge	–	1	1	10			
Extract particles (relion2.0)	g2.2xlarge	–	1	1	1	0.6	1.8	0.19
2D classification (relion2.0)	p2.8xlarge	8	1	17	1	5.4	41.8	1.6
Initial volume (eman2.12)	p2.8xlarge	–	1	16	1	0.08	0.66	0.03
3D classification (relion2.0)	p2.8xlarge	8	1	17	1	0.6	4.65	0.18
3D autorefine (relion2.0)	p2.8xlarge	8	1	17	1	0.7	5.53	0.22
Postprocessing (relion2.0)	p2.8xlarge	–	–	–	1	0.03	0.02	0
Total						44.1	157.54	57.78
FedCloud instance GPU								
Movie transfer (Aspera)	gpu1cpu6	–	–	–	2	24	–	–
Movie align (Motioncor2)	gpu1cpu6	1	–	–	2			
Ctf estimation (ctffind4)	gpu1cpu6	–	1	1	2			
Extract particles (relion2.0)	gpu1cpu6	–	1	1	2	0.43	–	–
2D classification (relion2.0)	gpu2cpu12	2	1	5	2	25	–	–
Initial volume (eman2.12)	gpu2cpu12	–	8	1	2	0.16	–	–
3D classification (relion2.0)	gpu2cpu12	2	1	3	2	2.13	–	–
3D autorefine (relion2.0)	gpu2cpu12	2	1	3	2	2.57	–	–
Postprocessing (relion2.0)	gpu2cpu12	–	–	–	2	0.01	–	–
Total						54.31	–	
Local instance GPU								
Movie align (Motioncor2)	titanxp	2	1	1	–	41	57.6	–
Ctf estimation (ctffind4)	titanxp	–	1	1	–			
Extract particles (relion2.0)	titanxp	–	1	24	–	0.42	0.6	–
2D classification (relion2.0)	titanxp	2	1	7	–	7.5	10.5	–
Initial volume (eman2.12)	titanxp	–	8	1	–	0.21	0.3	–
3D classification (relion2.0)	titanxp	2	1	7	–	1.28	1.8	–
3D autorefine (relion2.0)	titanxp	2	1	7	–	1.75	2.46	–
Postprocessing (relion2.0)	titanxp	–	–	–	–	0.003	0	–
Total						52.36	73.3	–

move the information to the cloud and which is the best way to do so?.

Regarding cloud configurations with GPU's, Relion 2.0 auto refine took 1.8 h when performed on local (two Titan XP) GPUs, as compared to 0.7 h in the Amazon Cloud (with eight Tesla K80 GPUs). It is certainly impossible to make an accurate comparison when the actual hardware was different in local and Cloud runs, but, in general, runs on the Cloud have a performance comparable to local runs. In short, and answering the first question, the overhead associated to Cloud computing for typical SPA workflows is not a limiting factor.

The second question is on easiness and cost of data transfer to and from the cloud. Naturally, movies represent large data sets, and the performance transferring them heavily depends on both network bandwidth and on the software used for this task. Simple and common solutions, like rsync, are notably outperformed by specialized software, as we clearly show in the Results section. Still, even using excellent solutions, over a 1 Gb/s link it takes approximately one day to transfer close to 1,000 movies of 30 frames of 8000×8000 pixels, totalling approximately 6 TB. To overcome this delay Scipion offers the possibility to run the first three workflow steps using the so-called 'Streaming mode', where movies are processed right after they are written on the disk. On the other hand, micrographs (more than an order of magnitude smaller) take just a couple of hours to transfer. Considering the above, it is clear that micrograph transfer is fast and easy, while movie transfer is not, suggesting that movie-based operations may be better handled locally, close to the microscope, starting the process on the Cloud after movie correction has taken place and they have been reduced to images/micrographs. Alternatively, the use of Scipion streaming mode greatly simplify preprocessing issues, since computation and transfer time are overlapped. It is also worth to note that moving Scipion projects across file systems and even different computers is easy, since Scipion projects are self-contained. Once movies have been corrected, there are no practical cloud data transfer bottlenecks for subsequent processing steps.

Completing the Discussion, the following subsections present a number of practical considerations related to the efficient use of cloud solutions.

3.1. Resource management

A key factor for an effective use of the Cloud is the analysis of the type of resources needed in each step, within the availability and price constraints of the Cloud IaaS provider.

CryoEM is a field that makes use of quite large data sets, especially at the start of the process, when videos are used. At the same time, the demands of the different processing steps of cryoEM workflows are quite heterogeneous. In general, memory (CPU and GPU memories) is the main limiting factor for many of the 3D operations, so that it defines the instance setup in practical terms. However, number crunching capabilities are more important than RAM for some other steps, a setup that is achieved more efficiently with a set of instances working cooperatively in a cluster and/or with GPU-powered instances.

GPU vector architecture allows for high parallel performance, although only for those programs carefully optimized for such architecture. The optimization step may have significant costs in developer man–hours, but processing time savings have led to an increase in the number of GPU-optimized versions of popular programs, like RELION-GPU (Kimanius et al., 2016, Motioncor2 (Zheng et al., 2017), GCTF (Zhang, 2016) and Ge-FREALIGN (Li et al., 2010). ScipionCloud includes GPU support for AWS and FedCloud.

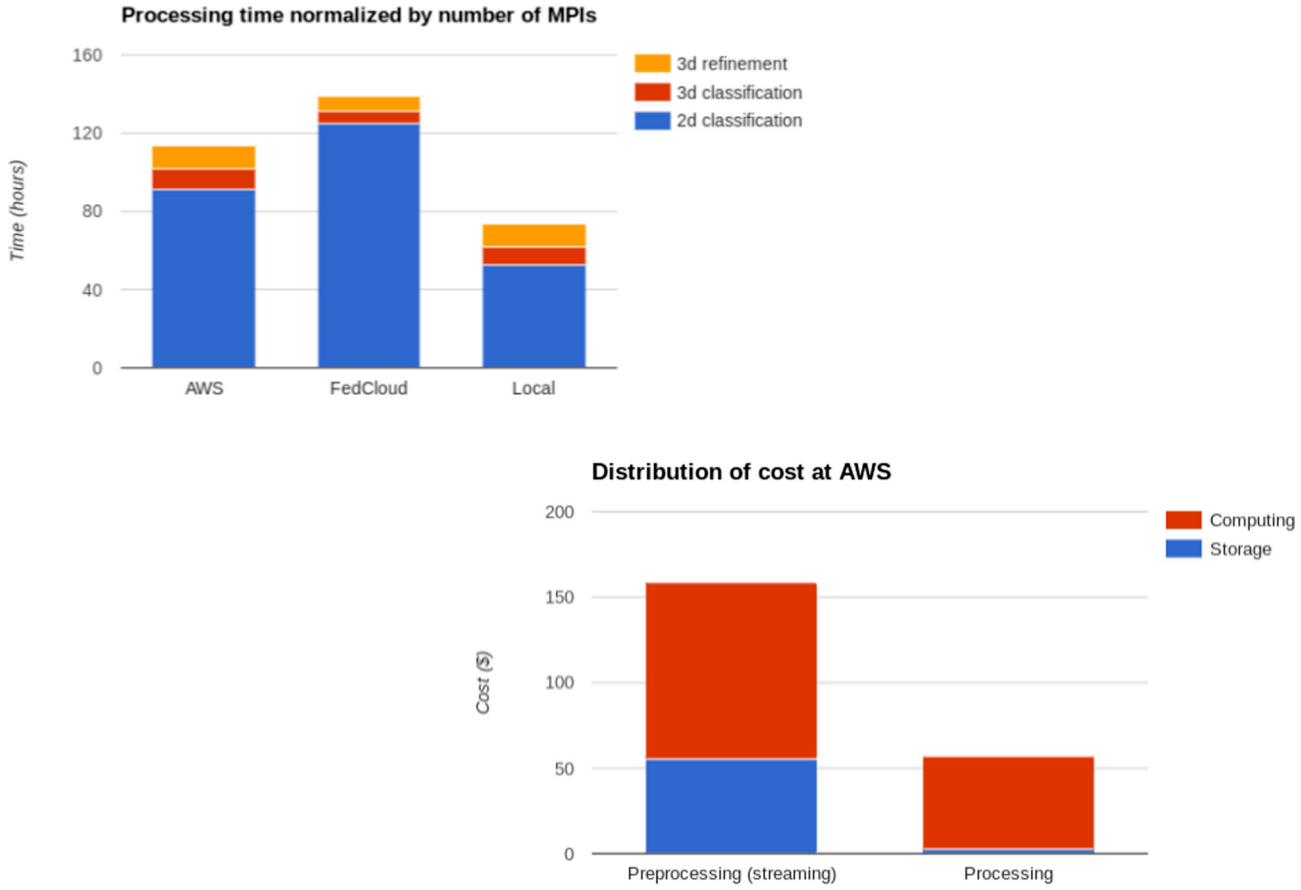

Fig. 2. Profiling summary and cost distribution. a) Chart of elapsed times normalized by MPI of the processing part for each profile. b) Chart with the distribution of financial on-demand costs for AWS.

Regarding other resources, like disk, we found that, in Amazon cloud, disk is relatively cheap, and it is only a limiting issue whenever movies are involved (another factor in favor of processing movies locally).

3.2. Cost efficiency

All hardware resources in cloud environments imply a cost proportional to its use. For example, memory is a key parameter of instances, and one that cannot be changed dynamically. Peaks in use of memory may be short, but the instance must have enough memory to handle them. The consequence under the pay-as-you-go model is that the user is paying for a large memory that is only sporadically used. Therefore, it is important to optimize wisely the resource allocation: making sure that active instances are productive, releasing instances no longer used, and dimensioning the instance resources (RAM, number of CPUs and GPUs, etc) and disk size according to the actual needs.

One additional question that arises is the convenience of processing on cloud (with a pay-per-use model) versus processing within a local facility (that typically is purchased and amortized).

For "short" projects, it is obviously cheaper to pay for effectively used resources than to buy a server. For extensive, all-year-long processing, the net cost of owning a server is cheaper than the corresponding Cloud bill, although Cloud cost is not disproportionate when all costs are considered. Let consider a concrete example. A standard server with 16 cores, 128 GB RAM, 2 GPU and 12 TB disk costs around 3,000$ per year (amortized over 3 years). In addition to this amortized acquisition cost, an appropriate fraction of a Full Time Equivalent System Manager (or similar) is required (2500$ per year, asuming a "standard" 50 K$ salary and a 5% dedication), plus housing in a datacenter (approx. 3 K$ per year per server) and electricity cost (approx. 2 k$ per year per server), elevating this cost to close to 10500 $ per year. Assuming an effective use of such a server of 80%, the cost per hour would be around 1.4$. Amazon *p2.xlarge* instances cost 0.702 $/h, but they are limited for EM processing (4 vCPU, 60 GB RAM, 1 GPU). Next model is *p2.8xlarge*, which climbs to 7.776 $/h, due to its impressive configuration: 32 vCPU, 488 GB RAM and 8 GPU.

Finally, architecture elasticity is a feature that only Cloud has, which may be very relevant to handle production peaks.

3.3. Conclusion

Our general conclusion is that with ScipionCloud, Cloud deployment is very easy and accessible, providing all Scipion advantages (ease of use, reproducibility, distributed computing) through a standard web browser. Regarding data transfer rates, typical academic network setups of 1 Gb/s bandwidth are perfectly adequate to handle the transfer of micrographs for several projects overnight; however, the transfer of movies is much slower, suggesting that they may be more effectively processed locally or using Scipion streaming mode. Note that in all cases the use of specialized software solutions for data transfer made a substantial difference in the good use of the available bandwidth. Additionally, the degree of interactivity in Graphical User Interfaces when changing from local settings to the cloud has also been brought to a level in which it is basically the same in the two environments. Finally, once effective solutions like ScipionCloud exists, the use of Cloud environments in cryoEM is basically an issue of the cost model a

given organization prefers to use, whether investing in resources and personnel or pay-per-use, and of the total money to be yearly spent depending on the specific use of the required resources.

4. Materials and methods

In this section we address the technical developments behind ScipionCloud and the profiling tests. First, we discuss the technical challenges and the solutions provided to them and, subsequently, we discuss on ScipionCloud availability.

4.1. Challenges porting Scipion to the Cloud

The single most critical challenge faced when developing ScipionCloud was related to the drastic change in the computing environment, from a static, local setting, to a dynamic and remote one. In such an environment, network delays start to be noticed; resources are inherently distributed, which adds additional concerns regarding data sharing and critical failures.

Network delays become most noticeable in interactive operations. This is why applications specifically developed from the start for cloud have an optimized web front-end that works well with very low network resources. In the case of Scipion and the EM packages it incorporates, most of their interfaces are based on X11 (www.x.org), which was developed for local area network operations. Therefore, under typical cloud environment restrictions (lower bandwidth, higher latency), those interfaces feel too slow for interactive use. We should note that this is not a problem specific to EM software, it appears in all X11-based programs.

Additionally, some applications display 3D graphics based on OpenGL (https://www.opengl.org). When such graphics are rendered locally, GPUs can be used to accelerate this rendering, hence providing real-time performance even with huge and complex volumes (indeed, that was the initial purpose of the GPU in graphic cards). In the cloud context, however, the rendering is done in the cloud instance itself, which implies that the acceleration has to occur at that instance as well. This latter requirement implies to have a GPU in the cloud instance and appropriate middleware to handle the particularities of this scenario.

At the same time, GPUs are used for actual computing. This translates into setting up GPU interface libraries and drivers, and enabling GPU code in the EM programs. Some programs can run CPU and GPU code from a single binary, while others require to manage two different sets of binaries, one for CPU code and another one for GPU-optimized code. In a local setting, these tasks are typically the responsibility of the user, but when porting to the cloud all these considerations have to be explicitly addressed when creating the cloud image. The same situation applies to many other details necessary for a positive user experience.

Regarding storage, a local computer typically has direct access to its disk, whether part of the computer itself or attached to an external disk subsystem. In cloud environments, however, this is not the case: the storage subsystem is virtualized and shared among multiple instances through the cloud internal network. This implies that the probability of losing access to any of the attached volumes is relatively high. Additionally, it must be ensured that only one instance performs the low-level access to such volumes, since concurrent access is handled at the higher file system level.

In a cluster setup there are three additional challenges: all nodes must be able to access the same data (while only one can actually access the underlying disk volume), the computing infrastructure has to be managed, and computing processes have to be distributed across the computing nodes. Again, relying on more virtualized elements increases the probability of failure.

Cloud infrastructures are very complex internally, a fact that has led to different evolution paths inside each cloud provider. There are standardization efforts for operating cloud infrastructures, but in this work we have found Federated Cloud and Amazon different enough to require specific images and initial setups (as explained in the Results section).

4.2. Technical solutions

4.2.1. Resource management

First we summarize how we proceeded to manage the resources in the cloud for the profiling tests.

For data storage we used volumes attached to the running instance: EBS volumes in Amazon, and Block Storage volumes in FedCloud. In both cases, the file system used was ext4. To simplify access to data volumes, ScipionCloud mounts them automatically on boot (provided that the external volume is already formatted).

ScipionCloud supports GPU acceleration, which is, indeed, an essential element in nowadays SPA software. As of May 2017, Amazon AWS top GPU-powered instances expose a Nvidia Tesla K80 GPU with 12 GB of VRAM, while in the market there are cards like Tesla P40 that provide 24 GB of VRAM. For Motioncor2, VRAM size is not an issue, but it may pose map size limits for GPU implementations of other programs, like Relion 2.0. Same happens with the cards offered by the Federated Cloud at the ISSAS cloud site, which currently are Nvidia Tesla K20 with only 4 GB VRAM.

Next, we summarize how we managed the instances to support all profiling scenarios, and how we supported distributed computing in the multiple-instance profiling.

For a prolonged use of cloud services, we found that it is beneficial to have some tool for managing the life cycle of the instances (create, start, stop). For single instances, we used EC2 console and occi client. For multiple instances (cluster setup), a more specialized toolkit is required. We used Starcluster on a desktop computer, but it can also run on a EC2 instance. We added a cluster template in the Starcluster configuration file in which we specified the image IDs, instance type and cluster size, the data volume and the spot bid. As cloud operation interfaces standardize, we expect that it will be possible to use a single tool to manage both single instance and cluster setups across all cloud platforms.

Regarding data sharing, we opted for NFS, an open-source file system that offers a good compromise of simplicity and performance and that simulates that the same files are locally available to all nodes, effectively encapsulating disk details in cluster setups.

In cryoEM, the de facto standard to coordinate the parallel processing across multiple nodes is MPI, which effectively encapsulates memory details. ScipionCloud integrates the Open MPI implementation of the standard.

4.2.2. Cost efficiency

In the case of AWS, a commercial service that offers a flexible pricing scheme, the researcher is billed for the costs of her processing. The billed cost will not only depend upon how many resources were used, but also on how they were requested. In this regard, Amazon offers three options: on-demand instances, spot instances and reserved instances. In the first option, the user requests resources for a specific time period and is billed monthly, according to her actual use. Spot instances can also be requested on demand, but their availability will depend on how much the user is willing to bid for them. Finally, for reserved instances, the user reserves them in advance for a minimum time period and is billed for the whole period, whether the instance was active or not. In the Results section we reported on-demand prices to provide a reliable, comparable reference of the total costs. If the researcher is willing to use spot instances, she can save 50% or even more. See

Sharma et al. (2016) for more information on spot instances and bidding.

In the profiling tests run on Amazon AWS, we recurred to on-demand instances to ensure that the workflow could run without interruptions. Our experience shows that is possible to use automatic bidding for days without disruptions.

Regarding FedCloud, its cost depends on research policies, and may involve project review, agreements with Virtual Organizations and/or, appropriate acknowledgements in scientific publications, among other factors. In our case, we could access EGI FedCloud resources at CESNET metacloud site and IISAS GPUCloud site thanks to the agreement they have with the enmr.eu Virtual Organization, to which we belong.

4.2.3. Interactive performance

In ScipionCloud we integrated Guacamole, which offers good performance and only requires a web browser on the user's computer. However, Guacamole currently handles only 2D acceleration, which means that 3D acceleration must be done on the server-side (that is, on the instance running on the cloud). For this release, we recurred to the non-accelerated OpenGL software rendering that MESA provides. Future releases will integrate the appropriate middleware to allow accelerated 3D OpenGL rendering. In the meantime, researchers can benefit from the Scipion architecture, that allows to perform the resource-demanding steps and 2D visualization in the cloud, and then use the OpenGL-based programs on their local computers.

4.2.4. ScipionCloud availability

Scientific computing environments require quite sophisticated system administration at different levels (operating system, libraries, applications), specially when using cluster architectures. We implemented all the required steps for such an environment in the ScipionCloud Linux-based image.

The image is available as a public AMI in AWS regions EU Ireland and US East named *ScipionCloud-1.1-beta*, and as a virtual appliance in EGI Applications Database (https://appdb.egi.eu/store/vappliance/scipion.v1.0).

It is possible to find Scipion source code and general documentation, as well as ScipionCloud-specific documentation, at Scipion project repository (https://github.com/I2PC/scipion/wiki/ScipionCloud).

Author contributions

JMC and PCM designed research. JGB designed the experimental workflow. JCA and LC developed the cloud image, administered cloud infrastructure and performed the profiling of workflows. JMRT supported the integration of Scipion in the cloud image. COSS and RM gave specific support to Scipion protocols and EM programs. JCA, LC and JMC prepared the manuscript. All authors reviewed the manuscript.

Acknowledgements

We would like to thank CESNET (https://www.cesnet.cz/), IISAS (http://ups.savba.sk/parcom/index.php) and IFCA (Institute of Physics of Cantabria, https://ifca.unican.es/en-us) staff for their kind technical support when using their sites on the EGI Federated Cloud.

The authors would like to acknowledge economical support from the Spanish Ministry of Economy and Competitiveness through Grants AIC-A-2011-0638, BIO2016-76400-R and BFU2016-74868-P. This work was co-funded by the European Union (EU) and Horizon 2020 through Grant West-Life (EINFRA-2015-1, Proposal: 675858) and the EGI-Engage project under Grant No. 654142, which allowed us to use the EGI Infrastructure. Work at Amazon AWS was funded by Instruct, part of the European Strategy Forum on Research Infrastructures (ESFRI), through grant reference RID:7 (Pilot EM Cloud Computing).

Appendix A. Supplementary data

Supplementary data associated with this article can be found, in the online version, at http://dx.doi.org/10.1016/j.jsb.2017.06.004.

References

Cianfrocco, M., Leschziner, A., 2015. Low cost, high performance processing of single particle cryo-electron microscopy data in the cloud. Elife 4.

de la Rosa-Trevín, J.E.A., 2016. Scipion: a software framework toward integration, reproducibility and validation in 3d electron microscopy. J. Struct. Biol. 195 (1), 93–99.

Fernández-del Castillo, E., Scardaci, D. Lopez, García, A., 2015. The egi federated cloud e-infrastructure. Procedia Comput. Sci. 68, 196–205.

Kimanius, D., Forsberg, B.O., Scheres, S., Lindahl, E., 2016. Accelerated cryo-em structure determination with parallelisation using gpus in relion-2. Elife 5.

Li, X., Grigorieff, N., Cheng, Y., 2010. Gpu-enabled frealign: accelerating single particle 3d reconstruction and refinement in fourier space on graphics processors. J. Struct. Biol 172 (3), 407–412.

Liao, M., Cao, E., Julius, D., Cheng, Y., 2013. Single particle electron microscopy: trends, issue and future perspective. Nature 504, 107–112.

Scheres, S.H., 2014. Beam-induced motion correction for sub-megadalton cryo-EM particles. Elife 3, e03665.

Sharma, P., Irwin, D., Shenoy, P., 2016. How not to bid the cloud. In: Proceedings of the 8th USENIX Workshop on Hot Topics in Cloud Computing (HotCloud 16).

Vinothkumar, K., Henderson, R., 2016. Single particle electron microscopy: trends, issue and future perspective. Quaterly Rev. Biophys. 49.

Zhang, K., 2016. Gctf: real-time ctf determination and correction. J. Struct. Biol 193 (1), 1–12.

Zheng, S., Palovcak, E., Armache, J.-P., Verba, K., Cheng, Y., Agard, A., 2017. MotionCor2: anisotropic correction of beam-induced motion for improved cryo-electron microscopy. Nat. Methods 14, 331–332.