electronics

MDPI

Article

A Web-Based Tool for Simulating Molecular Dynamics in Cloud Environments

Gonzalo Nicolas-Barreales [1], Aaron Sujar [1] and Alberto Sanchez [1,2,*]

[1] Department of Computer Science & Statistics, Universidad Rey Juan Carlos, 28933 Mostoles, Madrid, Spain; gonzalo.nicolas@urjc.es (G.N.-B.); aaron.sujar@urjc.es (A.S.)
[2] Research Center for Computational Simulation, Campus de Montegancedo/Scientific and Technological Park, 28660 Boadilla del Monte, Madrid, Spain
* Correspondence: alberto.sanchez@urjc.es

Abstract: Molecular dynamics simulations take advantage of supercomputing environments, e.g., to solve molecular systems composed of millions of atoms. Supercomputers are increasing their computing and memory power while they are becoming more complex with the introduction of Multi-GPU environments. Despite these capabilities, the molecular dynamics simulation is not an easy process. It requires properly preparing the simulation data and configuring the entire operation, e.g., installing and managing specific software packages to take advantage of the potential of Multi-GPU supercomputers. We propose a web-based tool that facilitates the management of molecular dynamics workflows to be used in combination with a multi-GPU cloud environment. The tool allows users to perform data pipeline and run the simulation in a cloud environment, even for those who are not specialized in the development of molecular dynamics simulators or cloud management.

Keywords: molecular dynamics; NAMD; multi-GPU; cloud; Amazon Web Services

Citation: Nicolas-Barreales, G.; Sujar, A.; Sanchez, A. A Web-Based Tool for Simulating Molecular Dynamics in Cloud Environments. *Electronics* **2021**, *10*, 185. https://doi.org/10.3390/electronics10020185

Received: 24 December 2020
Accepted: 13 January 2021
Published: 15 January 2021

1. Introduction

The main purpose of simulation is to shed light on the underlying mechanisms that control the behavior of a system. It is generally used to understand and predict the behavior of such a system. Predicting how the system will evolve and respond to its environment makes it possible to identify any necessary changes that will help make the system work as expected [1]. Mechanical simulation is important for multiple applications in different scenarios because of the advantages it offers over performing the real process [2]. The development of current simulation software is supported by supercomputing and cloud computing environments, which ensure, according to different approaches, the necessary processing, storage, and communication resources [3].

Molecular dynamics (MD) simulations make it possible to calculate the interaction of a large number of particles at the atomic level. These simulations are analyzed by a broad number of scientists from different fields, such as predicting protein structure, designing new drugs, or guessing binding affinity [4], to save costs in terms of materials, laboratory costs, and production time.

MD leads to the use of supercomputing environments considering the computing and memory cost of simulating millions of interactions in well-known biological molecules. MD simulators must be concerned with distributing computing and data while avoiding communication problems and maximizing the potential of supercomputers.

In the early 2000s, all the MD algorithms were based on Central Processing Units (CPUs) [5–7]. Those were oriented to use multiple CPUs to compute the simulation and usually relied on supercomputers where tens of CPUs performed the tasks. Recently, massively parallel computing has made a significant advance thanks to the introduction of the Graphics Processing Unit (GPUs). These are characterized by multiple parallel

operations in the dataset, which is especially suited to the needs of traditional MD. Multi-GPU environments represent the present in MD simulation, but new issues arise due to the difficulty of dividing and sharing the massive molecular constraints between computational resources [8].

Considering the above, users should face two concerns: (i) spend time and effort implementing those algorithms by installing or developing MD software packages and/or (ii) set up the environment (e.g., supercomputers, cloud services, personal computers) where they run the simulation. Note that end users may not have prior knowledge of MD simulation development or cloud management.

On the one hand, the execution of a MD simulation involves the preparation and processing of the molecule's data before launching the process. To do this, a molecular structure is usually downloaded from public databases, like [9]. Users must then perform three relevant tasks:

1. Establish a physical model, including the details of the atomic interaction.
2. Add a solution to the molecular system. Normally, the molecule is surrounded by water.
3. Equilibrate the whole system. Before carrying out the simulation, the molecular system needs to remain in a low-energy state after the introduction in a solution [10].

All this results in a complex workflow that is often tedious and involves downtime between tasks.

On the other hand, setting up the environment means that users must manage and configure the cloud environment. Cloud computing refers to both applications delivered as services over the Internet and the hardware and software systems in the data centers that provide those services [11]. This model of computing enables ubiquitous, convenient, on-demand access to a shared pool of configurable computing resources. Cloud computing has gained significant attention for running simulations in both the industrial and academic communities [12]. Different cloud platforms, such as Amazon Web Services, Google Cloud Platform and Microsoft Azure, allow end users to use all its capacity without the need to manage the system architecture. However, MD software packages require specific configuration in order to take full advantage of the hardware. Users should interact with the remote computer by commands and scripts to set up the simulation.

In this article, we present a web-based tool that allows end users to automate the previously described processes, both the molecular data pipeline and management and execution of the simulation in the cloud environment. Our tool provides an Infrastructure as a Service (IaaS) where users could manage the chosen cloud environment and set it up for MD purposes, and a Software as a Service (SaaS) where the application abstracts the whole workflow to make it transparent to the user. Using web technologies provides a friendly and recognizable UI where users can perform all MD tasks with a higher level of abstraction.

The paper is organized as follows. Section 2 analyzes different MD simulators and their ability to be used in cloud computing environments. Section 3 describes the main features of our proposal, from the MD simulation data pipeline to the creation of the infrastructure in the cloud environment. Section 4 presents how the tool works, carrying out a case study with the complex 4V4L molecular system. Lastly, Section 5 discusses the results, presents the conclusions drawn, and proposes future work.

2. Related Work

Molecular systems are composed of a large number of atomic particles that interact with each other. MD simulations [4] recreate these interactions by computational methods to estimate the behavior of a molecular system. Usually, systems are large enough not to be able to solve analytically.

MD considers the atoms' characteristics, and their force interactions, to accurately reproduce their movement over a time period. Discretization methods divide the simulation into time steps of a magnitude of femtoseconds (fs, 10^{-15} s) and compute interaction

forces and new atom positions using numerical integration. These calculations imply high computational and memory costs [8]. Several parallel computing algorithms have been proposed to accelerate interaction's calculation [13,14], resulting in multiple molecular dynamics simulators [15–19].

However, significant conformational changes still take too long when studying the dynamics of complex organic systems [20], which consist of millions of particles such as virus capsids, or even the entire virus. Usually, a desired biological process requires a high number of interactions that demands considerable simulation time and computing resources.

Even though most supercomputing centers are currently equipped with GPUs to perform massive calculations, this capability is very recent. Until a few years ago, supercomputing environments consisted of a large set of nodes, with multiple CPUs, interconnected among them. Proposed algorithms [5–7] were designed to take advantage of the parallelism provided by these types of environments, as the first versions of the well-known NAMD simulator [17]. Recently most of the current molecular dynamics simulators have been adapted to take advantage of the capabilities offered by modern GPUs to compute the interactions between atoms. NAMD [21] partitions the simulation of the molecular system into patches, which are distributed among the different computing nodes. GPUs are used as co-processors to perform the electrostatic forces calculations, while bonded forces are still calculated using CPUs [22]. GROMACS [18] carries out a spatial partitioning of the system to distribute molecular dynamics in multi-core architectures. The CPU uses the GPU as a co-processor to accelerate force calculations. In a similar way, AmberTools MD software packages (AMBER) [23] provides MD simulation in different computing environments (single/multi-CPU, single/multi-GPU) to perform a post hoc analysis. Instead, ACEMD [19] performs the calculation of all atomic forces on GPUs, using a single GPU for each force type. More recent works, such as MDScale [24], take advantage of massive MultiGPU environments enabling the division and exchange of atoms between GPUs.

If access to specific resources of a supercomputer is requested for MD, extensive bureaucratic procedures and knowledge of how to use it are required before running the simulation. In contrast, in most cases, the availability of cloud computing services for MD is immediate. Unfortunately, users must know how to manage the system and the infrastructure and be able to handle the requested resources, resulting in an additional workload. We found multiple tools that allow users to design and automate these processes. For example, Ref. [25] allows users to remotely simulate a cloud infrastructure to predict the cost of using specific software. Other software such as Aneka [26] enables the automatic deployment of a cloud infrastructure. The software provides an API, in which the users through the programming language .NET define the infrastructure and the software that runs on it. Despite the advantages, in both cases specialized knowledge of cloud services is required. Furthermore, the cost of configuring and deploying the cloud environment to run MD simulation software remains high.

The execution of a MD simulation requires a previous process of the molecular system. Typically, molecular data are obtained for general purposes and only provided with the names of the molecules and their positions. Additional data, such as the bonds or the charges of the atoms, are required for MD. It is therefore necessary to transform the existing molecular data, following a data pipeline or workflow, to obtain the necessary parameters to carry out the simulation. For instance, Purawat et al. [27] present a specific workflow using AMBER. This workflow focuses on the execution and analysis of the MD but leaves the responsibility for the construction of the system to the users. CHARMM-GUI [28] allows users to build and generate MD input files but does not provide job execution capabilities. VMD (Visual Molecular Dynamics) [29] complements some simulators (e.g., NAMD) and offers several plug-ins to perform multiple tasks. In particular, QwickMD plug-in [30] supplies a GUI to use NAMD more easily and simply. Knowing their functionality, users can combine different tools to perform MD simulation. Table 1 summarizes the software described throughout this section.

We propose a tool in which users can carry out the workflow in such a way that they intervene in the process as little as possible. Our approach is able to deploy all processes using cloud services in any cloud infrastructure. As far as we know, there is no software that fulfills all the necessary characteristics.

Table 1. Summary of MD simulators and cloud computing related software.

Software	Type	CPU/GPU
NAMD [21]	MD simulator	CPU and GPU
GROMACS [18]	MD simulator	CPU and GPU
ACEMD [19]	MD simulator	CPU and 1 GPU per force type
AMBER [23]	MD simulator and analisys	CPU and GPU
MDScale [24]	MD simulator	GPU (CPU only for execution flow)
Aneka [26]	Cloud Computing deployment	-
CloudSim [25]	Cloud Infrastructure Simulator	-
Kepler Workflow [27]	MD execution workflow	-
CHARMM-GUI [28]	MD input files generator	CPU
VMD [29]	MD visualization	CPU and GPU
QwickMD [30]	MD simulation GUI	-

3. Method

Generally, users need prior knowledge before performing MD simulation, not only about physical operation but also technical knowledge to be able to run on supercomputers and/or cloud environments. In many cases all this implies a high complexity and a long learning process. To simplify this process, we present a web-based application where users can—through a friendly UI, automatically and transparently—perform a data flow to get a molecular system ready and deal with a whole cloud infrastructure to run MD simulations. We have compiled the different steps needed to easily manage a molecular system and carry out a MD simulation. The application allows users to automate all processes using a remote computer environment.

The following subsections describe in detail the required pre-simulation molecular data processing and the integration mechanisms for running in cloud computing environments.

3.1. Molecular Data Processing

One of the first steps to perform MD is to obtain the data set to be simulated. Open access databases, such as Protein Data Bank (PDB) [9], offer a worldwide repository of structural data of large biological molecules. The information retrieved from PDB is incomplete because it is only a description of molecular systems, as it only provides the basic molecular data, such as the type of atoms and their positions. However, to carry out MD simulations, a pre-process is required to gather the additional molecule configuration needed. For this purpose we have grouped three relevant tasks that together form a work or data flow to prepare the molecular system and leave it ready for simulation. This data pipeline covers (see Figure 1) (i) generating a physical model, (ii) adding a solution, and (iii) thermal equilibration.

Generating the physical model is necessary to obtain specific atom's properties, such as their charges and the types of bonds of which they are part. In the usual molecular archives (e.g., from PBD), available biological structures only define the position and name of the atom, without any additional information. PSFGen plugin [31] for VMD compares and relates each atom of the molecular system with a series of topological description files. This process allows obtaining their charges and molecular bonds. Although PSFGen automates the task, it is highly demanding process; therefore, it would be advisable to run it in a cloud computing environment.

This result is not yet a realistic representation of the molecule because it is located in a vacuum. Adding a water solution to the molecular system is the next step in the data

flow. Solvate plugin [32] for VMD can perform this operation. It places multiple water molecules surrounding the system, resulting in a box-shaped molecular system.

This water box is inaccurate in physical terms because it has been created independently. This means that, considering the distance between atoms and their properties, the molecular system is in a high-energy state (i.e., unstable). The magnitude of this energy may not exist in the real world and therefore could not be simulated. Because of that, the target system must reach a lower energy state before simulation, which is achieved through a molecular equilibration process [33]. We have chosen the well-known NAMD molecular dynamics simulator [17], which offers a molecular system equilibration tool using the Root-Mean-Square Deviation (RMSD) method [34].

The resulting molecular structure, which can be used in any MD simulator, increases its size by about 40%. Depending on the molecular system, this size can vary from a few megabytes to thousands of gigabytes. For this reason, the described data pipeline implies a high cost of computing resources, and its execution fits perfectly in a cloud computing environment.

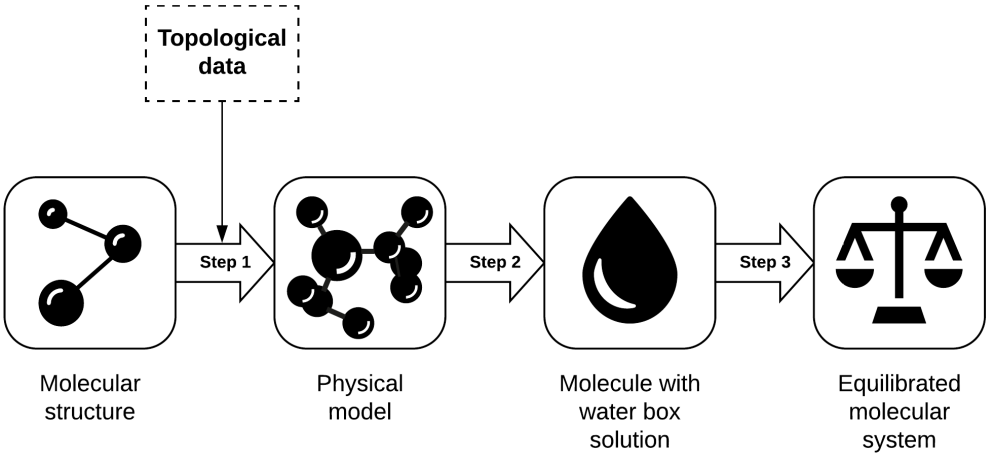


Figure 1. Data pipeline process. Before simulation, the molecular structure needs to be fully equilibrated.

3.2. Running MD in the Cloud

The main objective of our solution is to abstract the user from the resources needed to carry out the simulation. We propose that any process related to MD simulation runs directly in the Cloud. We have developed our tool as a web application, which provides a GPU-accelerated MD service running in cloud computing. Its aim is hiding the complexity of the use of a cloud infrastructure and the MD software. Figure 2 shows the proposed web application architecture. The main functionality is hosted on a web server that uses remote resources located in the cloud environment to perform the data pipeline shown above and the MD simulation. Since most of the computing workload falls on the cloud service, the web server can be placed internally on the user's system or on the cloud platform itself. Installation requirements are particularly low so that the web service can be supported by a simple AWS t2 instance [35], with only 1GB of RAM and a single CPU.

The server has a web service deployed, which controls the entire process. The web service acts as the communication link between the web application and the cloud infrastructure needed for simulation. The web service can be integrated with and manage the most common cloud computing environment, such as Microsoft Azure, Amazon Web Services, Google Cloud Computing, etc. All of these platforms have virtual machines with pre-set hardware (also called instances). They have CPU and GPU optimized instances, the latter even with single, multiple, or fractional GPUs, that can be used to enable a multi-GPU infrastructure for MD. The cloud platforms also provide storage services, which can be remotely accessed from the tool and the instances to store the molecular data and its corresponding metadata as cloud objects.

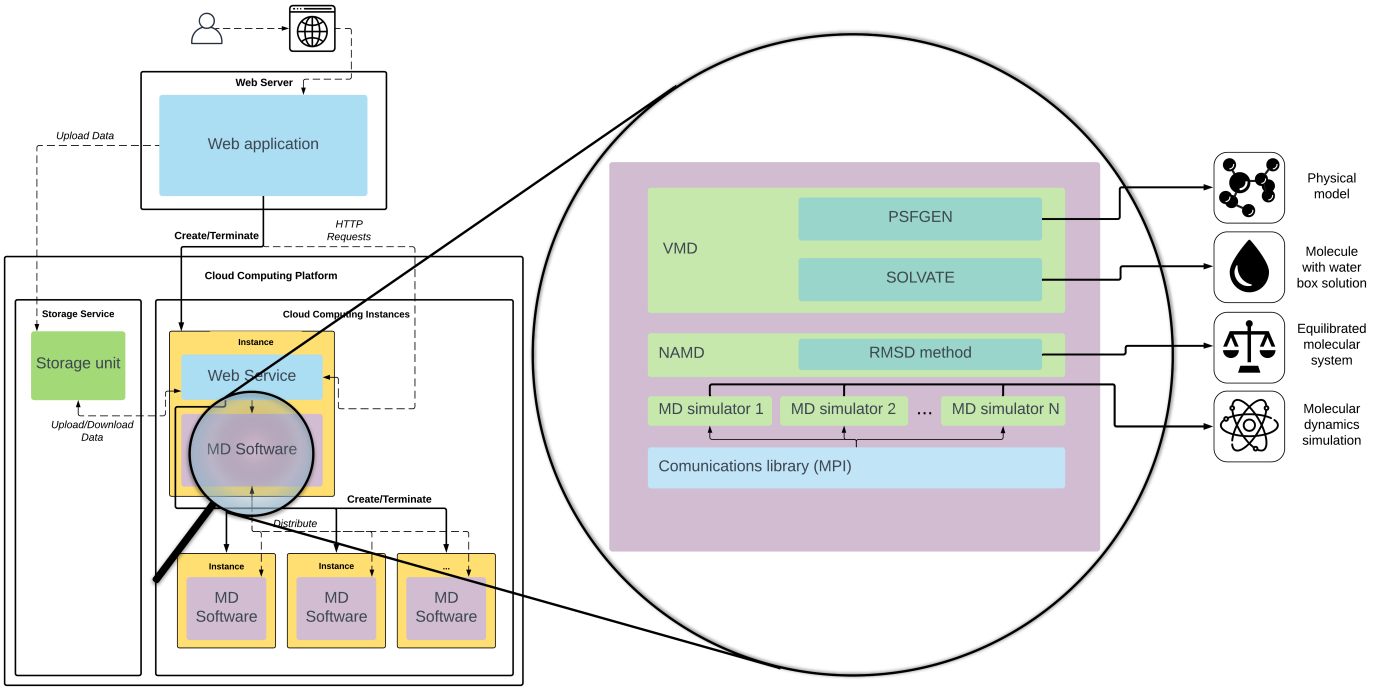


Figure 2. Cloud architecture diagram. Blue components belong to our proposed tool. In yellow are the custom instances that we have designed to run the MD software and communicate with each other and with our application. Green components belong to the cloud platform.

The web service is implemented as a REST API that allows the main application to make requests through the HTTP protocol, both to start or stop the execution of the software and to query its status. It also collects the data from the storage service and uploads the results obtained. The web service, like all the necessary additional software, is deployed in our own system images designed specifically to take advantage of the Multi-GPU cloud infrastructure. The images are based on a Linux operating system and contain the web service, VMD software with the PSFGen and Solvate plugins, NAMD with the RMSD algorithm, communication libraries, like MPI, and the necessary MD simulation software. The tool starts the instances according to the task to be performed, using only GPU-built instances when necessary. The main objective is that users are not concerned about how the cloud services for MD are started or finished when their run is over.

Our proposal automatically manages the following resources in the cloud platform:

- Users. User management allows each user to have their own space, where they can administer their access keys to the cloud platform. Access keys can be obtained directly from the control console of the cloud platform. Each key determines multiple permissions for the user in the cloud infrastructure.
- Storage. Since MD simulation requires and generates a large amount of information, we delegate the management of the files to the storage service of the cloud platform. In our tool, for each access key, several workspaces can be configured within the storage service, one for each molecular system to be considered. The tool uses each workspace to store all data related to the data pipe and simulation.
- Instances. As seen above, users may want to execute different stages of the data pipeline including simulation. When the execution of a stage is launched, the tool automatically deploys the requested instances into the cloud infrastructure by booting an own pre-configured system image with all the required MD software.
- Network communication. We manage the communication between the instances and the storage service in the deployment of the infrastructure. In case of simulation using several GPU-optimized instances at the same time, the tool manages the efficient

communication between all of them. Note that the access key must provide the corresponding permissions to be able to raise and communicate instances and services.

- Monitoring. Although complete cloud monitoring systems could have been used [36], we have chosen to provide the user only with information about the execution status in order to simplify their adaptation to the environment. These data are obtained directly from information provided by the cloud infrastructure itself. Each of the instances also executes a web service that communicates with the web server to let it know its execution status, which can be
    1. Preparing: data are being downloaded from the storage service.
    2. Running: the process is being executed.
    3. Finishing: the task has been completed successfully, and data are being uploaded to the storage service.
    4. Terminate: the task has been completed. This status triggers the tool to terminate the launched instances.
    5. Error: there has been an error in the execution of the process. The message is attached to a log. This status also triggers the tool to terminate the launched instances.

The defined architecture allows users to access our tool via a web browser (using smartphones, laptops, etc.) to easily run a MD simulation. As seen, the tool encapsulates the MD simulation functionality and cloud environment configuration, allowing users to perform the entire workflow through an easy-to-use user interface. All the simulation software runs on the cloud infrastructure, and users can leave the task unattended. This behavior approaches the definition of Software as a Service (SaaS).

Figure 3 shows the interaction between users and the proposed system. In essence, users interact with the application via a web interface. On the other side, the deployed web server is in charge of running a set of mechanisms related to cloud computing services. The execution process is as follows. First, the user configures the task to be performed in the application. The web server starts the required instance/s according to the configuration selected, using a pre-configured system image. Once the web server checks that communication has been established with the created instance/s, it sends a request with the script to run. The instance/s then download/s from the storage service all the data needed to perform the operation. When the task is completed, each instance uploads the results to the storage service, notifies the web server of its completion, and releases its resources. In case of an error, the instance returns an error message, attaching the log of the executed software. Finally, the web server notifies users via the web interface about the completion of the task.

The tool we propose is designed to use cloud computing services, reducing its cost as much as possible, avoiding unnecessary expenses if the execution fails or ends earlier than expected. Besides, the application allows users to leave it unattended when MD tasks, as usual, take a long time to complete. The tool is fault-tolerant against failures or disconnections of the web server. At each step of the simulation process, the tool stores the execution status and the ID of the instances. This means that if the web server is disconnected, it is possible to recover the execution status without losing progress. Errors that occur during the execution of the software are also managed. In case of failure, the application receives the corresponding status of the instance, which contains the log data of the software that was running. This log is shown to the users via the web interface.

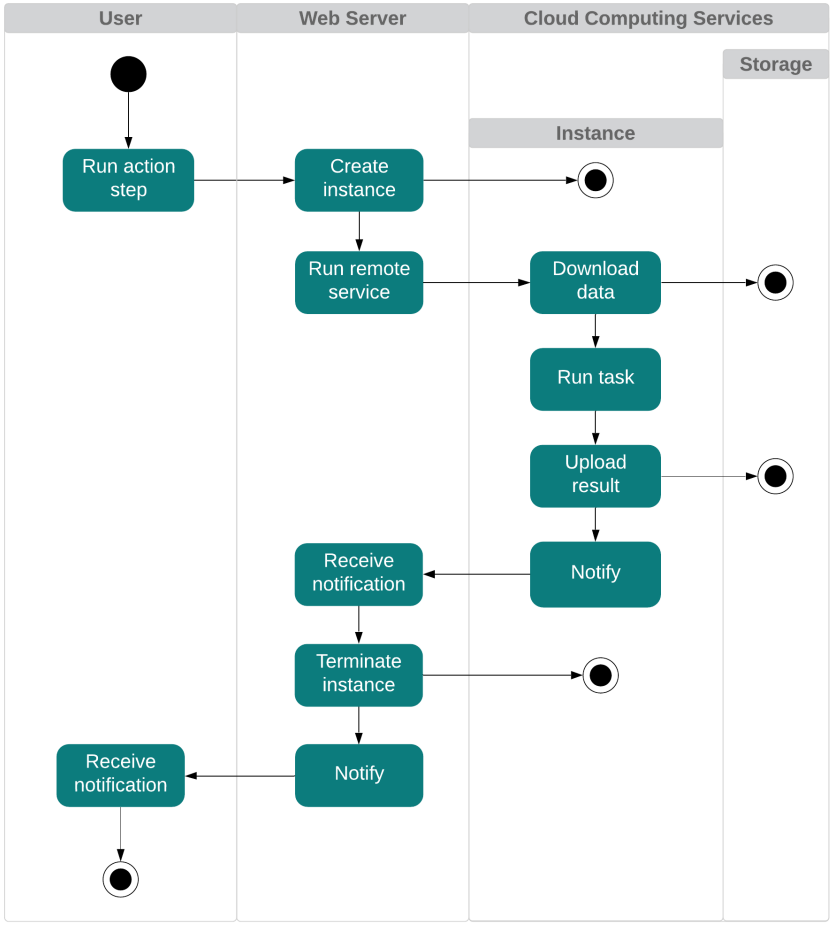


Figure 3. Workflow of the tool's steps, including from the user's request to the completion of the task.

4. Results

In this section, we present a case study deploying our solution in a local environment and using Amazon Web Services (AWS SDK [37]) as a cloud platform test. Although we have developed the software for this cloud platform, it can be easily migrated to other environments, such as Microsoft Azure or Google Cloud Platform, using their corresponding SDK. The web server would be adapted to manage the virtual machines using the corresponding SDK. Additionally, new system images should be created with the MD software.

As a molecular system we have chosen the molecule 4V4L as an example of a complex system downloaded from PDB (see Figure 4). The 4V4L molecular system obtained from PDB [38] is composed of about 128,000 atoms reaching up to 500,000 atoms at the end of the process. The development of the data pipeline to build the molecular system to be simulated implies a RAM memory cost of 15 gigabytes. In this case, this amount can be easily achieved with a basic AWS Elastic Compute Cloud (EC2) CPU-based instance, like t2.2xlarge [35]). Larger molecular systems, such as 3J3Q [39], can reach up to 65 million atoms, requiring about 3 terabytes of RAM and necessarily involving the need to use supercomputers or powerful cloud computing environments. For example, x1e.32xlarge AWS EC2 CPU-based instance [40] provides 3.9 terabytes of RAM, enough to solve this problem, but it would also be possible to use t2.2xlarge EC2 instances with an additional 3 terabyte hard disk used as swap. The latter configuration is much cheaper but involves a much longer runtime.

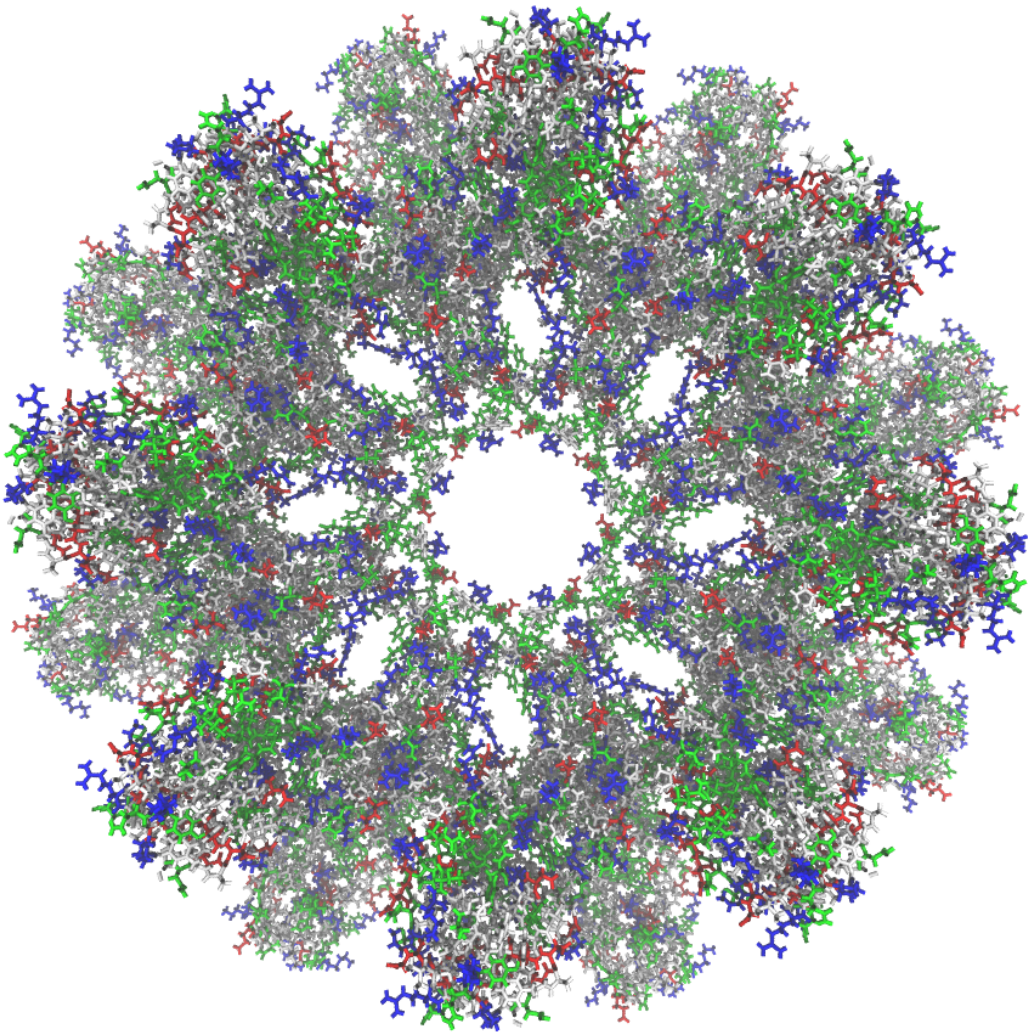


Figure 4. 4V4L molecular structure from Protein Data Bank. The image has been generated using VMD software.

For equilibration and simulation processes it is strongly recommended to use AWS EC2 instances that provide GPUs due to the high computational workload of these operations. For instance, p2.16xlarge AWS EC2 GPU-optimized instance [41] enables up to 16 Nvidia Tesla K80 GPUs. Note that the simulation of larger molecular systems may require even more GPUs. In a regular environment it would be necessary to properly configure the network communication between the different instances to have a functional Multi-GPU environment, besides installing and managing the necessary software. Our web-based approach initializes, deploys instances with the required number of GPUs, and enables communication among them without user intervention.

As a cloud object storage service, we used AWS Simple Cloud Storage Service (S3), which provides scalability and data availability. In the AWS nomenclature, a bucket is similar to a folder in the storage service that stores objects, which consist of data, in this case molecular systems, and its descriptive metadata.

Figure 5 shows the complete architecture of the test bed running the AWS infrastructure required for the deployment and use of the application. In the next paragraphs, we describe the entire procedure from the user management to the execution of MD simulation going through the workflow using our tool. Note that it is not necessary to carry out all the steps in the process. It is possible to load the result of some of them, and users can also rerun each step without having to restart the whole process.

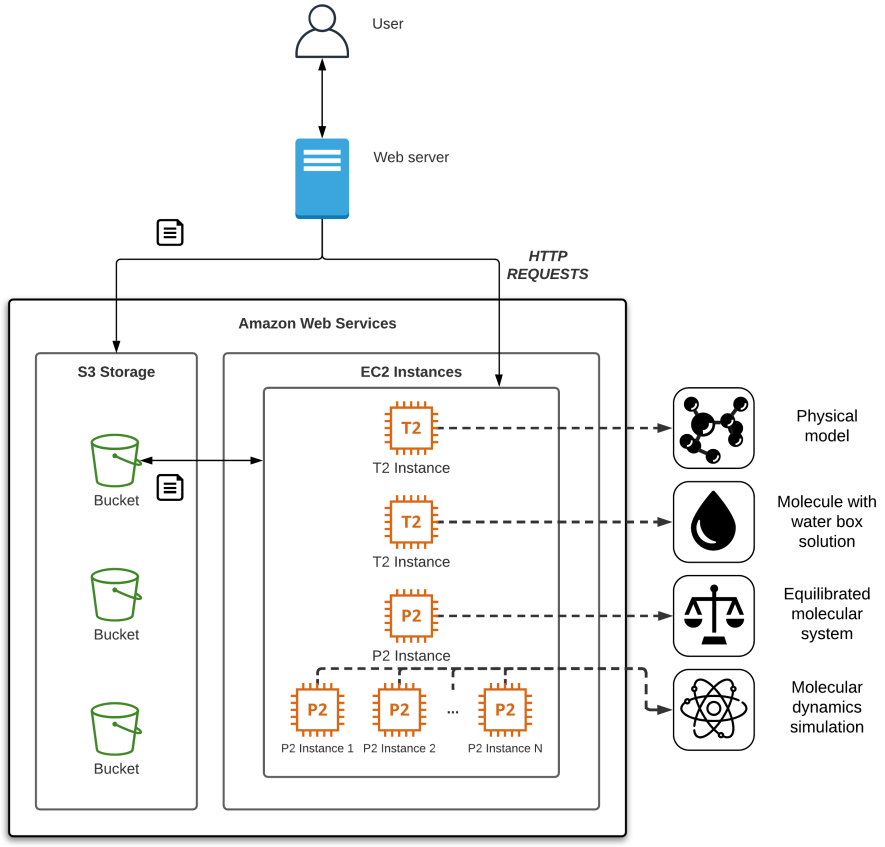


Figure 5. Detailed diagram of the testbed architecture based on Figure 2 using the AWS cloud computing environment.

The first step is to register access to the cloud platform. User data are encrypted and stored in a SQL database on the web server. Specifically, users will be asked for the credentials they want to use on the specific cloud platform, in this case AWS. Users must have an AWS account and have generated the external access keys in the AWS Identity and Access Management (IAM) service. Note that the cloud AWS user must have permissions to remotely manage EC2 instances and store data in S3. Access keys consist of two parts: an access key ID and a secret access key. Each access key is stored and assigned to the logged-in user as shown in Figure 6. Once users log in the tool, they can start managing their work environment.

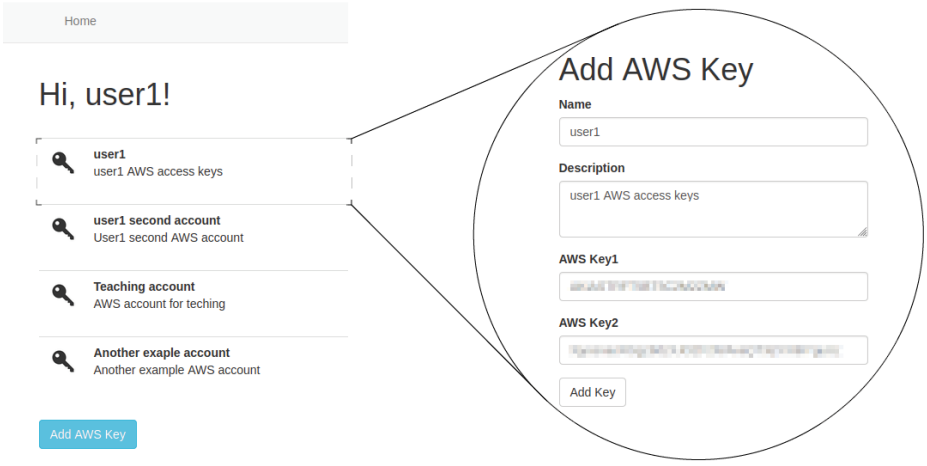
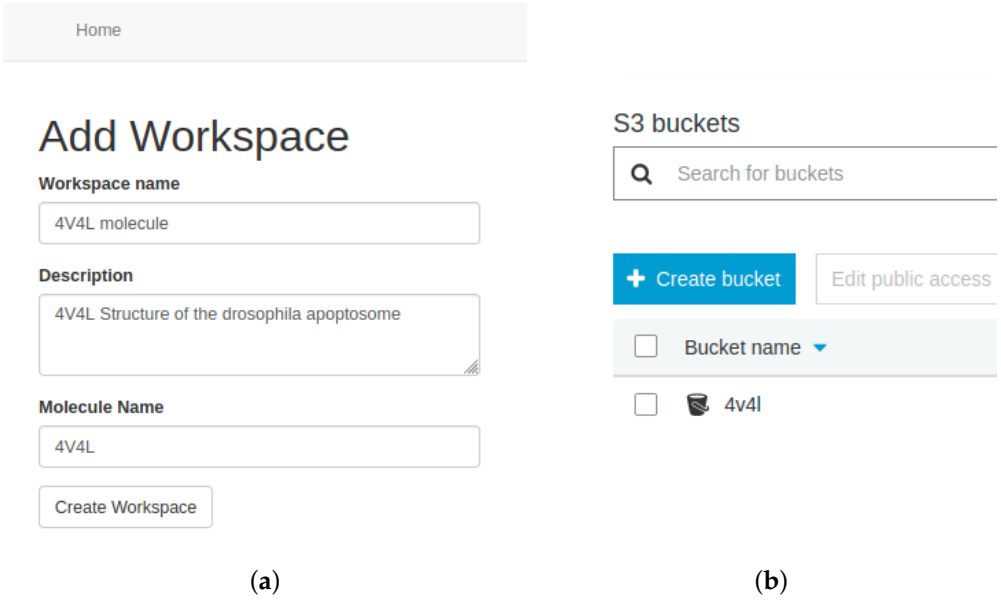


Figure 6. Access key association. On the left side, each user can store several access keys to log in to their cloud account. On the right, the image shows the detail of an example AWS access key.

At this point, users must configure a workspace associated with their account, which represents a bucket within the AWS S3 storage service. It is recommended to create a single workspace per molecular system to be simulated. The bucket stores all data related to the selected molecular system. EC2 instances will access this bucket to retrieve the data and upload the results of the MD process. Figure 7 shows a workspace created in the tool, along with its representation in S3.



(**a**) (**b**)

Figure 7. Creation of a workspace for a molecular system: (**a**) shows how a workspace is created in the tool; (**b**) depicts the related bucket created in the AWS S3 service.

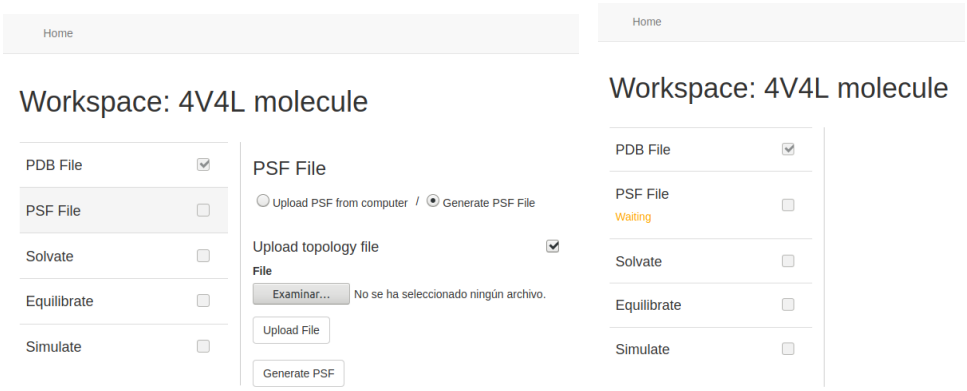
The next step is to select the data of the molecular system to work with. To simplify the process, our tool offers two options to upload a file to the cloud storage service: choose a local file with the information or provide a link to gather it directly from PDB, as shown in Figure 8.

(**a**) (**b**)

Figure 8. Molecular system loading process: (**a**) shows how a link is provided from the PDB; (**b**) shows the PDB's 4V4L molecular system.

Once molecular data have been uploaded, users can start the data pipeline depending on their needs. The first stage they can run consists of relating the atoms to the molecule to which they belong, assigning their bonds and electrical charges. Users can upload and use a specific topology file, but if not, they can carry out that calculation in a single AWS EC2 t2 instance using the VMD software [29], which compares molecular data with different existing topologies. The tool performs everything related to the management of this instance. Users thus only have to configure the execution and the startup request.

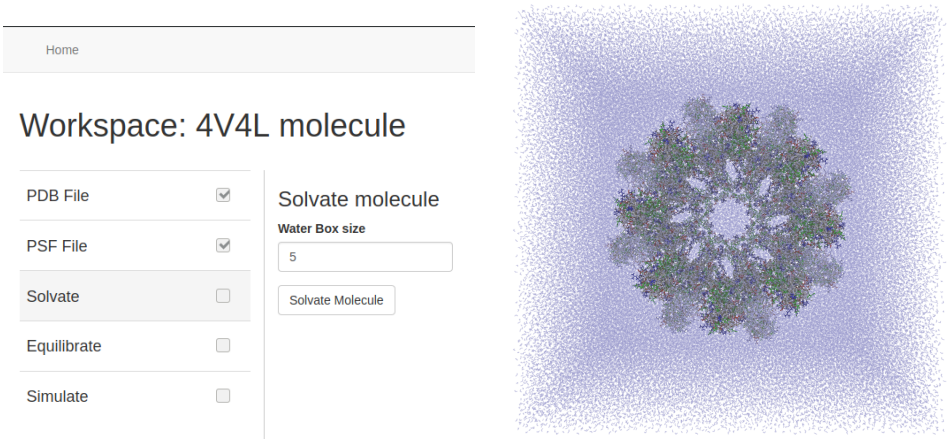
The status of the stage execution can be consulted in the application, which is retrieved directly from the own EC2 instance. Figure 9 depicts how the tool is configured to create a Protein Structure File (PSF). PSF files contain all the molecular information that is not included in the previously obtained PDB data, such as bonds or atom charges. They should be used together with the PDB file to simulate MD.



(**a**) (**b**)

Figure 9. PSF creation stage: (**a**) shows the detail of the tool with the configuration options. Users can choose default topologies or upload a custom topology file. (**b**) The tool's interface when the execution of this step is in progress.

As described in Section 3.1, a molecular system cannot be located in the vacuum and needs to be inside of a water solution. The second stage consists of creating a molecular structure that forms a cubic system composed of the original data surrounded by multiple water molecules (see Figure 10). The size of the structure depends on the magnitude of the original system, but users can set how large the margin will be. VMD software also performs this process on a CPU-based t2 instance of the cloud infrastructure. Just like the previous stage, users select some parameters, and the tool manages the whole process.



(**a**) (**b**)

Figure 10. Stage of adding water solution: (**a**) shows a snapshot where users specify the solvation box's size. In (**b**), a 4V4L molecular system is inside of a solvation box.

Introducing water into the system makes it unstable for MD simulation. To solve this we use NAMD software, which implements an option to balance molecular systems. As in the previous stages, the tool manages the execution of this process in a single EC2 GPU-optimized p2 instance. Users only have to select the relevant parameters to perform a

small simulation to look for the low-energy state. Our tool allows users to choose multiple settings like simulation temperature, time step (in fs), simulation steps, and step reset speed values. Figure 11 shows an example of the configuration of this stage.

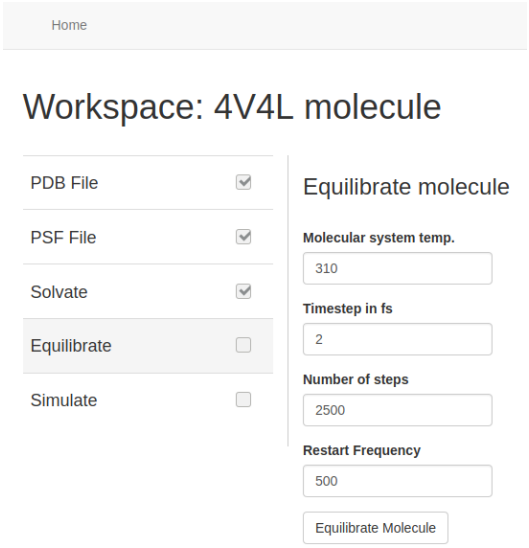


Figure 11. Stage of equilibration of the molecular system. Users can set the relevant parameters in the balancing process.

With the previous stages the molecule would be ready to be simulated. Finally, our tool offers to run the simulation using different MD simulators. We have configured an AWS EC2 instance, which can be replicated in a cloud environment by connecting between its copies, by installing NAMD and our own implemented Multi-GPU version of [8], named as MDScale [24]. As future work we propose to include other MD simulators, such as GROMACS or AMBER. In any case, we allow users, if they wish, to use their own system image (called Amazon Machine Image in AWS) to carry out the simulation process using custom software. To do this, users must also provide the script needed to run the simulator (see Figure 12).

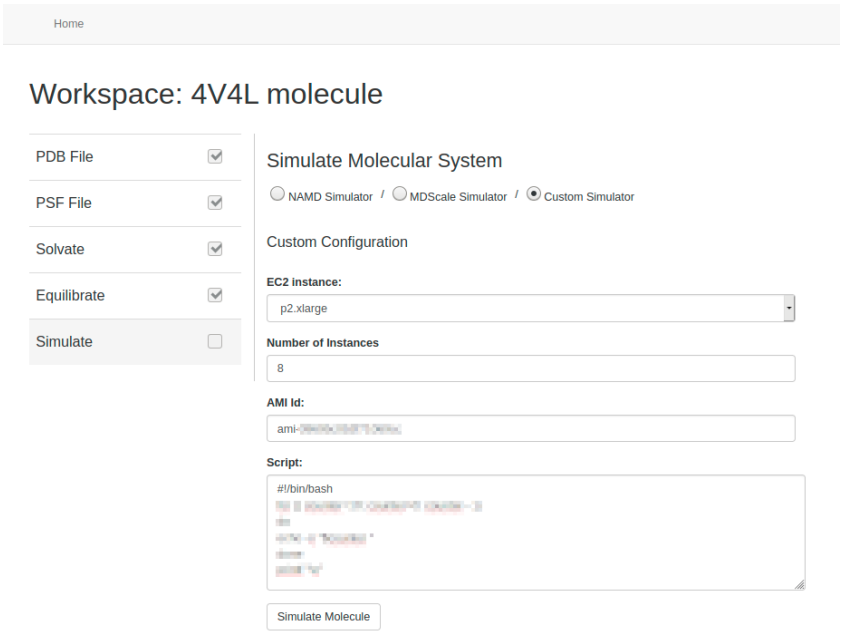


Figure 12. Setting up a custom MD simulation. Users can provide a custom AMI, the type and number of instances, and an execution script.

Users can set the basic parameters related to the simulation such as temperature, time-step, or output writing frequency. Depending on the MD simulator used, they can also set other optional parameters such as steps per cycle, cutoff radius and frequency for the calculation of the short-range electrostatic forces, cell-list update frequency, and/or the use of specific algorithms such as PME [42] or MSM [43] to calculate the long-range electrostatic forces (see Figures 13 and 14). In addition, users can choose the number of GPUs, specifically Nvidia Tesla K80 in AWS, deployed for simulation. Our tool automates the initialization of the required EC2 p2 instances based on the number of GPUs requested and manages the network communication among them. At the end, the resulting data from the molecular system are stored in the AWS S3 storage service.

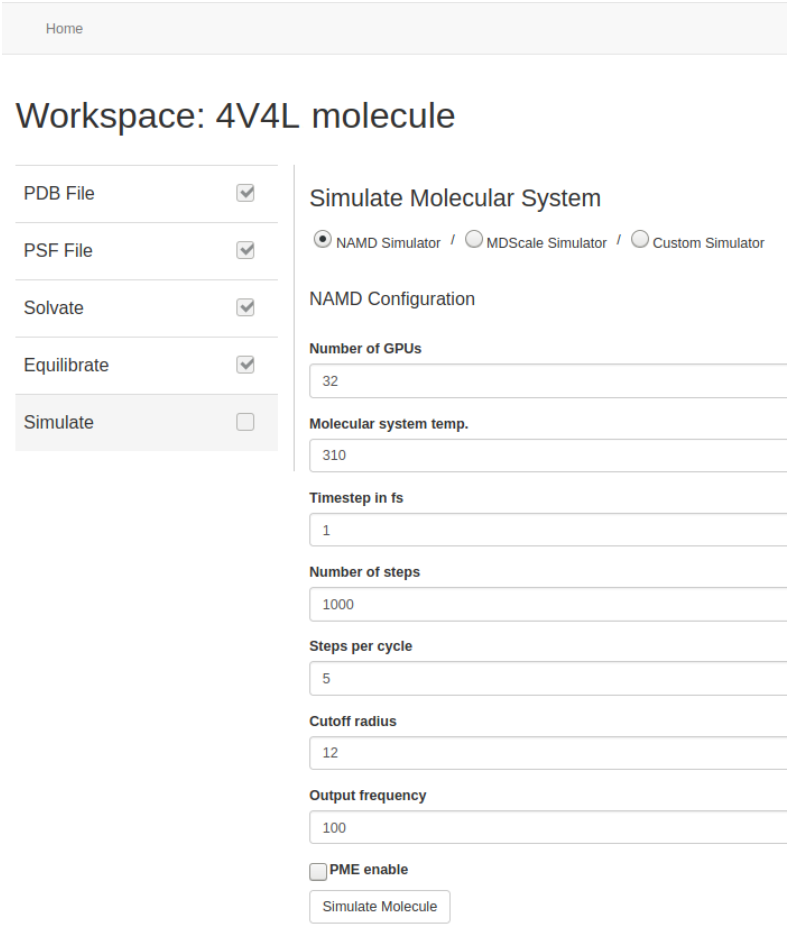


Figure 13. Configuration of the MD simulation using NAMD. Users can specify the details of the simulation and choose the number of GPUs to be used.

Finally, we carried out the entire simulation process using the 4V4L molecular system in NAMD and MDScale to demonstrate the possibilities of our tool. Table 2 shows the execution times for each step of the workflow, from the initial data processing to the simulation of the molecular system. The deployment and execution time for cloud environments depend on the resources required and their availability at any given time; therefore, there is not much emphasis on this matter. Thanks to the possibility of configuring the simulation, users are able to carry out balanced tests with the different types of atomic forces.

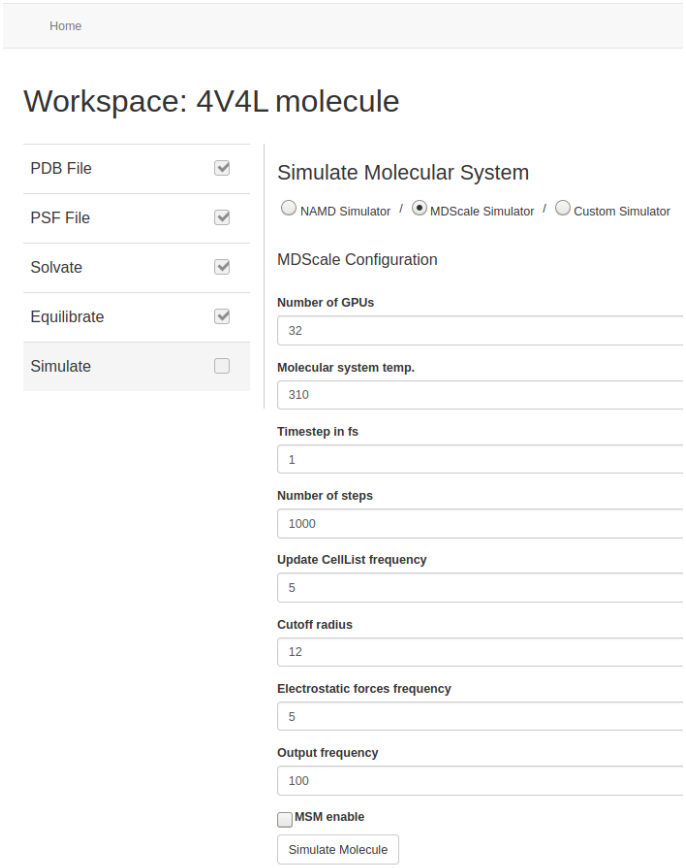


Figure 14. Configuration of the MD simulation in MDScale. Users can specify the details of the simulation and choose the number of GPUs to be used.

Table 2. Runtime of each workflow step, differentiating between the deployment time of EC2 instances and the execution time of the MD simulator. MD simulation was performed using 8 GPUs, while the equilibration process was run on only 1 GPU. All other steps do not require the use of GPUs.

	EC2 Deploy	Execution Time
PSF generation	85.6 s	47.9 s
Solvation	69.2 s	58.5 s
Equilibrium	153.2 s	3334.3 s
Simulation (NAMD)	304.4 s	295.7 s
Simulation (MDScale)	283.7 s	237.5 s

5. Conclusions

In this paper, we have presented a web-based application that facilitates MD simulation using cloud computing services and automates related tasks. This application allows users to manage both data preparation and simulation of molecular systems in a Multi-GPU cloud computing environment. Through a friendly user interface, our system provides an easy way to perform GPU-accelerated MD simulation even for non-specialist users.

We designed the tool in order to save time in setting up a cloud environment and cost managing the cloud resources efficiently to prepare molecular systems and run MD simulators. The main target of this paper was to discuss the functionality of the tool in a cloud computing environment.

Our solution relies on a typical web application structure, which consists of a web server and a web service following the REST API architecture. Users interact with the simulation through a web browser, run any task, and leave it unattended. The application can stop any cloud resource to avoid unnecessary cost.

Our tool has been designed to facilitate the use of different MD software. We installed and compiled tools like VMD, NAMD, etc., in a cloud instance to perform the MD workflow without the need to understand how to configure and execute them. Furthermore, this design is open to changes. Any software or algorithm can be updated, added, or replaced in order to perform any task of MD simulation process. This would enable users to generate comparisons between new algorithms and techniques that will be developed in the future.

The code of the tool is freely available for use (http://monkey.etsii.urjc.es/md/easymd). We expect our tool can be used in the future for anybody interested in MD simulation. This application could also be extended to create other solutions used a cloud environment. Just by changing the instances and slightly modifying the web service, this tool can be easily adjusted to perform different tasks that require cloud computing.

Our tool has some limitations in supporting cloud environments. We designed the application with Amazon Web Services in mind, due to the possibility of using GPU instances. However, we have intended to offer support for other cloud environments with different configurations and SDK, such as Google Cloud Platform or Microsoft Azure, or even for use in custom local clusters or supercomputing centers.

As future work, we propose to implement security mechanisms to establish communication between the web service and the infrastructure, providing also protection against DoS, SQL injection, and similar attacks. In addition to the simulation process, rendering of molecular dynamic images is also a lengthy and costly process. Rendering can also be performed within the cloud computing environment, and this feature will be explored in future work.

Author Contributions: Conceptualization, A.S. (Alberto Sanchez); software, G.N.-B. and A.S. (Aaron Sujar); investigation, A.S. (Alberto Sanchez) and G.N.-B.; resources, A.S. (Alberto Sanchez); writing—original draft preparation, A.S. (Aaron Sujar) and G.N.-B.; writing—review and editing, A.S. (Alberto Sanchez), A.S. (Aaron Sujar) and G.N.-B.; project administration, A.S. (Alberto Sanchez); funding acquisition, A.S. (Alberto Sanchez). All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the Spanish Ministry of Science, Innovation and Universities grant number RTI2018-098694-B-I00.

Acknowledgments: This work has been partly supported by the Spanish Ministry of Science, Innovation and Universities (grant RTI2018-098694-B-I00). The evaluation was made possible by a grant from AWS Cloud Credits for Research.

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

The following abbreviations are used in this manuscript:

MD	Molecular dynamics
CPU	Central Processing Units
GPU	Graphics Processing Unit
IaaS	Infrastructure as a Service
SaaS	Software as a Service
NAMD	Not Another Molecular Dynamics Program
GROMACS	GROningen MAchine for Chemical Simulations
ACEMD	Acellera Molecular Dynamics
AMBER	AmberTools MD software packages
VMD	Visual Molecular Dynamics
PDB	Protein DataBank
RMSD	Root-Mean-Square Deviation method
AWS	Amazon WebServices
IAM	Identityand Access Management
API	Application Programming Interface
REST	Representational State Transfer

References

1. Mefteh, W. Simulation-Based Design: Overview about related works. *Math. Comput. Simul.* **2018**, *152*, 81–97. [CrossRef]
2. Winsberg, E.; Gruner, S. Science in the Age of Computer Simulation. *Minds. Mach.* **2013**, *23*, 251–254. [CrossRef]
3. Savaglio, C.; Campisano, G.; di Fatta, G.; Fortino, G. IoT Services Deployment over Edge vs Cloud Systems: A Simulation-based Analysis. In Proceedings of the IEEE INFOCOM 2019—IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS), Paris, France, 29 April–2 May 2019; pp. 554–559. [CrossRef]
4. Schlick, T. *Molecular Modeling and Simulation: An Interdisciplinary Guide*; Springer: Secaucus, NJ, USA, 2002.
5. Plimpton, S.; Heffelfinger, G. Scalable parallel molecular dynamics on MIMD supercomputers. In Proceedings of the Scalable High Performance Computing Conference SHPCC-92, Williamsburg, VA, USA, 26–29 April 1992; pp. 246–251.
6. Eleftheriou, M.; Fitch, B.G.; Rayshubskiy, A.; Ward, T.J.C.; Germain, R.S. Scalable framework for 3D FFTs on the Blue Gene/L supercomputer: Implementation and early performance measurements. *IBM J. Res. Dev.* **2005**, *49*, 457–464. [CrossRef]
7. Vadali, R.V.; Shi, Y.; Kumar, S.; Kale, L.V.; Tuckerman, M.E.; Martyna, G.J. Scalable fine-grained parallelization of plane-wave–based ab initio molecular dynamics for large supercomputers. *J. Comput. Chem.* **2004**, *25*, 2006–2022. [CrossRef] [PubMed]
8. Novalbos, M.; Gonzalez, J.; Otaduy, M.A.; Martinez-Benito, R.; Sanchez, A. Scalable On-Board Multi-GPU Simulation of Long-Range Molecular Dynamics. In Proceedings of the Euro-Par 2014 Parallel Processing, Porto, Portugal, 1 December 2014; pp. 752–763. [CrossRef]
9. Berman, H.M.; Westbrook, J.; Feng, Z.; Gilliland, G.; Bhat, T.N.; Weissig, H.; Shindyalov, I.N.; Bourne, P.E. The Protein Data Bank. *Nucleic Acids Res.* **2000**, *28*, 235–242. [CrossRef]
10. Gallo, M.T.; Grant, B.J.; Teodoro, M.L.; Melton, J.; Cieplak, P.; Phillips, G.N., Jr.; Stec, B. Novel procedure for thermal equilibration in molecular dynamics simulation. *Mol. Simul.* **2009**, *35*, 349–357. [CrossRef]
11. Armbrust, M.; Fox, A.; Griffith, R.; Joseph, A.; Katz, R.; Konwinski, A.; Lee, G.; Patterson, D.; Rabkin, A.; Stoica, I.; et al. A View of Cloud Computing. *Commun. ACM* **2010**, *53*, 50–58. [CrossRef]
12. Zhao, W.; Peng, Y.; Xie, F.; Dai, Z. Modeling and simulation of cloud computing: A review. In Proceedings of the 2012 IEEE Asia Pacific Cloud Computing Congress (APCloudCC), Shenzhen, China, 14–17 November 2012; pp. 20–24. [CrossRef]
13. Kupka, S. Molecular dynamics on graphics accelerators. In *Proceedings of CESCG*; Institute of Computer Graphics and Algorithms: Castá-Papiernicka Centre, Castá, Slovakia, 2006; pp. 1–4.
14. Yang, J.; Wang, Y.; Chen, Y. GPU accelerated molecular dynamics simulation of thermal conductivities. *J. Comput. Phys.* **2007**, *221*, 799–804. [CrossRef]
15. Brooks, B.R.; Brooks, C.L., III; Mackerell, A.D., Jr.; Nilsson, L.; Petrella, R.J.; Roux, B.; Won, Y.; Archontis, G.; Bartels, C.; Boresch, S.; et al. CHARMM: The biomolecular simulation program. *J. Comput. Chem.* **2009**, *30*, 1545–1614. [CrossRef]
16. Pearlman, D.A.; Case, D.A.; Caldwell, J.W.; Ross, W.S.; Cheatham, T.E.; DeBolt, S.; Ferguson, D.; Seibel, G.; Kollman, P. AMBER, a package of computer programs for applying molecular mechanics, normal mode analysis, molecular dynamics and free energy calculations to simulate the structural and energetic properties of molecules. *Comput. Phys. Commun.* **1995**, *91*, 1–41. [CrossRef]
17. Phillips, J.C.; Braun, R.; Wang, W.; Gumbart, J.; Tajkhorshid, E.; Villa, E.; Chipot, C.; Skeel, R.D.; Kalé, L.; Schulten, K. Scalable molecular dynamics with NAMD. *J. Comput. Chem.* **2005**, *26*, 1781–1802. [CrossRef]
18. Abraham, M.J.; Murtola, T.; Schulz, R.; Páll, S.; Smith, J.C.; Hess, B.; Lindahl, E. GROMACS: High performance molecular simulations through multi-level parallelism from laptops to supercomputers. *SoftwareX* **2015**, *1–2*, 19–25. [CrossRef]
19. Harvey, M.J.; Giupponi, G.; Fabritiis, G.D. ACEMD: Accelerating Biomolecular Dynamics in the Microsecond Time Scale. *J. Chem. Theory Comput.* **2009**, *5*, 1632–1639. [CrossRef] [PubMed]
20. Dreher, M.; Piuzzi, M.; Turki, A.; Chavent, M.; Baaden, M.; Férey, N.; Limet, S.; Raffin, B.; Robert, S. Interactive Molecular Dynamics: Scaling up to Large Systems. *Procedia Comput. Sci.* **2013**, *18*, 20–29. [CrossRef]
21. Rodrigues, C.; Hardy, D.; Stone, J.; Schulten, K.; Hwu, W.m. GPU acceleration of cutoff pair potentials for molecular modeling applications. In Proceedings of the 2008 Conference on Computing Frontiers, CF'08, Ischia, Italy, 8–10 May 2008; pp. 273–282. [CrossRef]
22. Phillips, J.C.; Stone, J.E.; Schulten, K. Adapting a message-driven parallel application to GPU-accelerated clusters. In Proceedings of the SC '08: 2008 ACM/IEEE Conference on Supercomputing, Austin, TX, USA, 13–19 November 2008; pp. 1–9. [CrossRef]
23. Salomon-Ferrer, R.; Case, D.A.; Walker, R.C. An overview of the Amber biomolecular simulation package. *WIREs Comput. Mol. Sci.* **2013**, *3*, 198–210. [CrossRef]
24. Nicolas-Barreales, G.; Novalbos, M.; Otaduy, M.A.; Sanchez, A. A Prototype of a Scalable Multi-GPU Molecular Dynamics Simulator for Large Molecular Systems. In *Spanish Computer Graphics Conference (CEIG)*; Garcia-Fernandez, I., Ureña, C., Eds.; The Eurographics Association: Goslar, Germany, 2018; pp. 25–28. [CrossRef]
25. Calheiros, R.N.; Ranjan, R.; Beloglazov, A.; De Rose, C.A.F.; Buyya, R. CloudSim: A toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms. *Software Pract. Exp.* **2011**, *41*, 23–50. [CrossRef]
26. Vecchiola, C.; Chu, X.; Buyya, R. Aneka: A Software Platform for .NET-based Cloud Computing. *High Speed Large Scale Sci. Comput.* **2009**, *18*, 267–295.
27. Purawat, S.; Ieong, P.; Malmstrom, R.; Chan, G.; Yeung, A.; Walker, R.; Altintas, I.; Amaro, R. A Kepler Workflow Tool for Reproducible AMBER GPU Molecular Dynamics. *Biophys. J.* **2017**, *112*. [CrossRef]
28. Jo, S.; Kim, T.; Iyer, V.G.; Im, W. CHARMM-GUI: A web-based graphical user interface for CHARMM. *J. Comput. Chem.* **2008**, *29*, 1859–1865. [CrossRef]

29. Humphrey, W.; Dalke, A.; Schulten, K. VMD—Visual Molecular Dynamics. *J. Mol. Graph.* **1996**, *14*, 33–38. [CrossRef]
30. Ribeiro, J.; Bernardi, R.; Rudack, T.; Stone, J.; Phillips, J.; Freddolino, P.; Schulten, K. QwikMD—Integrative Molecular Dynamics Toolkit for Novices and Experts. *Sci. Rep.* **2016**, *6*, 26536. [CrossRef] [PubMed]
31. PSFGen Plugin for VMD. Available online: https://www.ks.uiuc.edu/Research/vmd/plugins/psfgen/ (accessed on 12 June 2020).
32. Solvate Plugin for VMD. Available online: https://www.ks.uiuc.edu/Research/vmd/plugins/solvate/ (accessed on 12 June 2020).
33. Brooks, B.R.; Bruccoleri, R.E.; Olafson, B.D.; States, D.J.; Swaminathan, S.; Karplus, M. CHARMM: A program for macromolecular energy, minimization, and dynamics calculations. *J. Comput. Chem.* **1983**, *4*, 187–217. [CrossRef]
34. Coutsias, E.A.; Wester, M.J. RMSD and Symmetry. *J. Comput. Chem.* **2019**, *40*, 1496–1508. [CrossRef]
35. Amazon EC2 T2 Instances. Available online: https://aws.amazon.com/ec2/instance-types/t2/ (accessed on 12 June 2020).
36. Montes, J.; Sanchez, A.; Memishi, B.; Pérez, M.S.; Antoniou, G. GMonE: A complete approach to cloud monitoring. *Future Gener. Comput. Syst.* **2013**, *29*, 2026–2040. [CrossRef]
37. AWS SDK for Python (boto3). Available online: https://aws.amazon.com/sdk-for-python/ (accessed on 12 June 2020).
38. Yuan, S.; Topf, M.; Dorstyn, L.; Kumar, S.; Ludtke, S.J.; Akey, C.W. PDB ID: 4V4L. Structure of the Drosophila apoptosome at 6.9 angstrom resolution. *Structure* **2011**, *19*, 128–140. [CrossRef] [PubMed]
39. Perilla, J.R.; Zhao, G.; Zhang, P.; Schulten, K.J. PDB ID: 3J3Q. Atomic-Level Structure of the Entire HIV-1 Capsid. *Nature* **2013**, *497*, 643–646. [CrossRef]
40. Amazon EC2 X1e Instances. Available online: https://aws.amazon.com/ec2/instance-types/x1e/ (accessed on 12 June 2020).
41. Amazon EC2 P2 Instances. Available online: https://aws.amazon.com/ec2/instance-types/p2/ (accessed on 12 June 2020).
42. Darden, T.; York, D.; Pedersen, L. Particle mesh Ewald: An Nlog(N) method for Ewald sums in large systems. *J. Chem. Phys.* **1993**, *98*, 10089–10092. [CrossRef]
43. Hardy, D.; Stone, J.; Schulten, K. Multilevel Summation of Electrostatic Potentials Using Graphics Processing Units. *Parallel Comput.* **2009**, *35*, 164–177. [CrossRef]