CONCURRENCY AND COMPUTATION: PRACTICE AND EXPERIENCE
Concurrency Computat.: Pract. Exper. 2015; **00**:1–26
Published online in Wiley InterScience (www.interscience.wiley.com). DOI: 10.1002/cpe

Multi-layered simulations at the heart of workflow enactment on Clouds

Simon Ostermann[1]*, Gabor Kecskemeti[2] and Radu Prodan[1]

[1] *Institute of Computer Science, University of Innsbruck, 6020 Innsbruck, Technikerstr. 21a, Austria*
[2] *Institute for Computer Science and Control, Hungarian Academy of Sciences, 1111 Budapest, Kende u. 13-17, Hungary*

SUMMARY

Scientific workflow systems face new challenges when supporting Cloud computing, as the information on the state of the used infrastructures is much less detailed than before. Thus, organising virtual infrastructures in a way that not only supports the workflow execution, but also optimises it for several service level objectives (e.g., maximum energy consumption limit, cost, reliability, availability) become reliant on good Cloud modelling and prediction information. While simulators were successfully aiding research on such workflow management systems, the currently available Cloud related simulation toolkits suffer form several issues (e.g., scalability, narrow scope) that hinder their applicability. To address these issues, this article introduces techniques for unifying two existing simulation toolkits by first analysing the problems with the current simulators, and then by illustrating the problems faced by workflow systems. We use for this purpose the example of the ASKALON environment, a scientific workflow composition and execution tool for Cloud and Grid environments. We illustrate the advantages of a workflow system with directly integrated simulation back-end and how the unification of the selected simulators does not affect the overall workflow execution simulation performance. Copyright © 2015 John Wiley & Sons, Ltd.

Received ...

KEY WORDS: Scientific workflow, Cloud computing, simulation toolkits

1. INTRODUCTION

Scientific workflows [1] enable constructing and executing large scale distributed applications based on well understood basic building blocks, designed for scientists with less expertise in organising and enacting a complex application. The burden of organisation and enactment lies on the underlying workflow management systems, that must not only ensure the proper and timely execution of the users' complex applications, but should also optimise their distribution and schedule on the available infrastructures. With the advent of Cloud computing [2], workflow

*Correspondence to: Institute of Computer Science, University of Innsbruck, 6020 Innsbruck, Technikerstr. 21a, Austria. E-mail: simon@dps.uibk.ac.at

management systems must not only cope with the available infrastructures, but must also be able to decide when and how to improve user experience with the inclusion of leased virtual infrastructures.

Although the building blocks of these scientific workflow applications could have execution time in the range of months, their enaction by the workflow systems could have significant effects both on their runtime as well as on the underlying infrastructures [3]. In the past, several workflow systems used simulators [4, 5] to evaluate the possible effects of particular enactment scenarios on workflows and infrastructures. Simulations are important tools to speed up research evaluations that otherwise would need too much time in reality. The increase in speed is normally reached by simplifying the model of the system to be simulated trying to stay as close to reality as possible. Simulations in some extreme cases are increasing evaluation speed to such levels that they allow close to real-time evaluation of multiple situations. Unfortunately, past workflow management techniques, which were incorporating simulators in their decision making process, hardly considered the highly volatile and dynamic nature of Cloud systems.

Although several Cloud simulators exist today (Cloudsim, GroudSim, iCanCloud) [6, 7, 8], they can hardly support the requirements of current workflow management systems. They are frequently oriented towards the Cloud user, therefore mostly considering Clouds as a black box. Unfortunately, this behaviour does not allow the analysis of infrastructure level effects of the various decisions made by workflow management systems. Even in such cases, when a simulator offers insights on how Clouds internally operate, they are mostly focused on specific areas (e.g. providing accurate CPU or network sharing, energy modelling) while neglecting others; therefore they restrict the use cases in which these simulators would be useful for the complex decision making process [9] in Cloud aware workflow management systems.

Through the example of a well researched scientific workflow management system (namely ASKALON [10]), we analyse in this paper the possible improvements one could gain by integrating a user-side simulator (called GroudSim [7]) with an internal infrastructure focused simulator (called DISSECT-CF, DIScrete event baSed Energy Consumption simulaTor for Clouds and Federations) [11]. Using this approach, we can not only fulfil the demands of current research directions, but also allow the widening of research applied in scientific workflow management systems. Thus, this paper has two distinct contributions: (i) the integration of two complete simulators in a way that keeps their features while minimising the overhead caused by their joint operation, (ii) the analysis of new research directions the merged simulators could offer to the community researchers responsible for scientific workflow management systems.

We have chosen the ASKALON system because it has already been integrated with the GroudSim [7] simulator as a support tool in its workflow enactment-related decisions. For the role of the second simulator, we have selected the DISSECT-CF versatile simulation framework, as it is capable to simulate the internals of Cloud infrastructures allowing the evaluation of energy consumption, network behaviour and the effects of multi-tenancy. Although we have evaluated the integration on these specific systems, our carefully executed extensions show that the introduced techniques would be applicable to similar workflow systems too [12]. Our extensions show that an existing workflow system could already benefit from such integrated simulations with minimal or no changes to its workflow management techniques. The combination of ASKALON and the integrated GroudSim and DISSECT-CF simulators come along with many improvements which directly or

DOI: 10.1002/cpe

indirectly influence the accuracy of the simulation results of scientific workflow applications in ASKALON.

We show in this article how DISSECT-CF was integrated into GroudSim and evaluate the resulting performance loss. Compared to the more accurate results and added possibilities the performance is still reasonable. Additional workflow simulation experiments are done using ASKALON and the combined simulators to show the functionality and the negligible overheads due to the new simulation components.

The rest of the paper is organised as follows. In Section 2 we review the currently existing workflow systems and describe the advantages of our approach. In Section 3 we present the background information required to understand the existing systems in isolation, followed by details about the integration in Section 4. We summarise the new possibilities achieved by this extension in Section 5 and conclude the paper with a short outlook into upcoming work in Section 6.

2. RELATED WORK

Scientific applications are complex systems consisting of different programs that often need days or weeks to be executed. Users of such applications apply the workflow paradigm or other techniques to build larger scale applications out of existing programs, increase programming productivity and parallelism, and achieve faster execution times on. To research the impact of different schedules or optimisations, simulators are often employed to reduce the time between implementation of features and their verification.

The surveys in [4, 5] give a good overview of existing simulators, some of them covering the field of Cloud computing. The status of GroudSim in the [4] survey shows important missing features, while some features must have been overlooked by the authors (a cost model exists in GroudSim since its initial version, and has been extended over the years to support all commercially available billing models). Other crucial features provided by DISSECT-CF on the internals of infrastructure Clouds (e.g. energy models, more complex networking) are introduced in this publication and have been added to GroudSim too.

GridSim [13] and its extension CloudSim [6] are well-known simulation environments for task executions on Grid and Cloud platforms. As our previous work showed, the scalability and flexibility is their biggest problem [7].

iCanCloud [8] is a new contribution to the area of Cloud simulators specialised on Amazon EC2 resources using a configuration GUI. Because of its user orientation, iCanCloud lacks crucial functionality needed for Green IT research [14] such as power consumption, and has no workflow support.

WorkflowSim [15] is an open source workflow simulator that extends CloudSim by providing new constructs for simple management and simulation of workflows. It models workflows as a DAG and provides out of the box implementation for several popular workflow schedulers (e.g., HEFT, Min-Min) and task clustering algorithms. Its main disadvantage (alongside its limitation to the DAG model lacking loops) is that it misses a connection to a real-life workflow management system such as ASKALON.

DOI: 10.1002/cpe

SimGrid [16] has been developed over the past years as a versatile, accurate and scalable simulator. Compared to our solution, it lacks support for dynamic workflow applications, as it only supports static DAGs, and does not offer important features like real-life and simulated executions within the same environment. Researching new methods and ideas needs therefore twice effort required in ASKALON: first the validation must be performed in SimGrid, and afterwards the new code needs to be rewritten for the real execution environment.

The previews integration work of GroudSim into ASKALON [17] showed the usefulness of such a integrated approach that other workflow systems such as Pegasus [18], Taverna [19] or WS-PGRADE [20] lack. With the integration of DISSECT-CF, all features added to GroudSim are also automatic available in ASKALON, allowing better simulations leading to more accurate and realistic research results.

Compared to other existing simulators, two features make the combination of the ASKALON-GroudSim system with DISSECT-CF unique: (i) the possibility to simulate and execute workflow applications directly within the same environment, and (ii) the integration of a unified power utilisation and resource sharing model for simulating data centre components.

3. BACKGROUND

Simulation is a known useful practice when trying to solve complex problems like scheduling, resource management or workflow executions. There are multiple tools available for this purpose, as mentioned in section 2, but they either lack functionality or are not user friendly. Especially the high interest in power-aware methods is not satisfactory with the current available simulators in the scope of workflow executions on Cloud resources. To overcome this drawback, we developed a simulator specialised on Infrastructure as a Service (IaaS) Clouds with focus on power consumption and scalability of the simulation. We integrated this simulator into an existing framework for workflow development, execution and simulation called ASKALON.

3.1. ASKALON

ASKALON, an existing middleware researched at the University of Innsbruck, provides an integrated environment to support the development, simulation and execution of scientific workflows on dynamic Grid and Cloud infrastructures [10]. Figure 1 shows the design of the ASKALON system with focus on the integrated simulator, explained in detail in Section **??**. Workflows can be graphically programmed in an abstract and user-friendly fashion in a platform independent Java application. The abstraction is used to shields the users from the low-level Cloud infrastructure technology details as no such knowledge is needed in the workflow creation process. Workflows can be created in a "drap and drop" fashion form existing abstract activities where only the input and output port must be connected to each other to build the workflow structure. Once the workflow is created and confirms to the model checker, it can be submitted for execution to the ASKALON services, which allow for long lasting executions in online interactive or offline batch mode. A command line client allows script-based batch execution of the workflows in an XML language representation for cron-job based executions of single or multiple workflows. Execution information is stored in a database allowing online and post-mortem analysis using a graphical

Prepared using cpeauth.cls
DOI: 10.1002/cpe

performance measurement tool or custom SQL-queries. The three main components that handle the execution of workflows are explained in the following.

Execution Engine. The execution engine (EE) is responsible for processing the workflow, unrolling the parallel loops into executable tasks and their management. Submission of jobs and transfer of data to the compute resources is done with a suitable protocol such as `ssh` and `scp` for Cloud resources or GRAM and GRIDFTP in a Globus/Grid environment. For simulated workflow executions, we developed a new provider for the Globus CoG-kit [21] that allows the use of the existing abstraction model to interact with the integrated simulator. There is also a scheduling module included in the EE2 that allows an easy integration of existing and new scheduling algorithms (e.g. MOHEFT [22], HEFT [23], MCT [24]).

GridARM/GLARE. The resource manager has the task to manage existing Grid resources or to provision the correct amount of Cloud resources at the correct moment to allow the EE run the workflow as decided by the scheduler. The scheduler can therefore include precise information in the task mappings (i.e. run the task on a specific resource from a Cloud provider) or less restrictive mappings (i.e. run a task on any resource of Cloud provider). To request and release Cloud instances, the resource manager communicates with different Cloud providers, or in the simulation case with GroudSim, to provision Cloud instances using predefined images for the required applications. The applications may be automatically deployed on the instance after its boot or the image may already include the desired applications.

GAB. The GroudSim-ASKALON-Bridge is responsible for distinguishing between the components of ASKALON that can only be effectively run in real a real environment, and the simulated execution environment operated by GroudSim. As ASKALON is used for executions on real hardware, this module is needed to allow the integration of the simulator in a transparent fashion to the other components that are not simulated such as the scheduler, resource manager and job or file transfer submissions. This module ensures that the simulation time is only advanced when no more new events from ASKALON are generated to avoid increased simulated time due delayed job submissions. As the EE is heavily using threads for pipelining and parallelism in the processing the workflow structure and execution, we aim not to stop it more then needed. Therefore, most EE threads continue working while other functions are blocked using advanced lightweight Java synchronization mechanisms (`BlockingQueues`), as they have to wait for the simulated results. This enables a better performance of the system, but does not allow to clearly identify the overhead added with this synchronization mechanism as most threads are not blocked. Adding blocking mechanisms to all EE threads would allow to better measure GAB overheads but this would significantly decrease the overall performance meaning that EE threads need to support pause functions. Therefore, we decided our design for performance in order not to add additional overheads for making this mechanism better measurable.

DOI: 10.1002/cpe

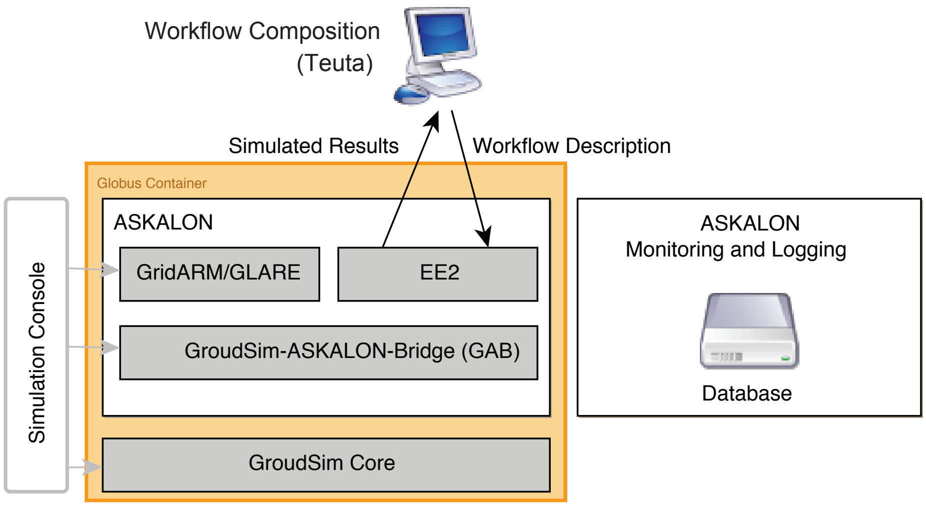

Figure 1. The architecture of ASKALON

3.2. GroudSim

GroudSim is an event-based simulation toolkit for scientific applications running on combined Grid and Cloud infrastructures developed in Java[†]. GroudSim uses a discrete-event simulation toolkit that consists of a future event list and a time advance algorithm that offers improved performance and scalability compared to other process-based approaches used in related work [25]. The simulator can be used in a stand-alone fashion or integrated in the ASKALON environment. The later option allows seamless development, debugging, simulation and execution processes using the same ASKALON interface offered to the end-users.

Figure 2 shows the most important components of GroudSim, which collaborate internally and act as interfaces to communicate with the GAB. The two central parts of the simulation framework are: (i) the *event system* storing information about the type and time of the events, and (ii) the *simulation engine* responsible triggering the events at well-defined time instances. Events can simulate job executions, file transfers, availability of resources (including failures), and background load. The other GroudSim core components are:

Resource module that manages the simulated resources and communicates them to GridARM/GLARE;

Synchronisation module which allows synchronisation of the simulation time and the time used by the EE. When the EE is generating new tasks and submits them to the simulator, the simulator must wait until all current tasks are submitted before the simulation time can be advanced;

Background loader adds additional load to the resources upon requests from the user or GAB. The load can be achieved by using traces from the Grid Workload Archive [26] or by using synthetic job distribution functions;

[†] http://www.dps.uibk.ac.at/projects/groudsim/

DOI: 10.1002/cpe

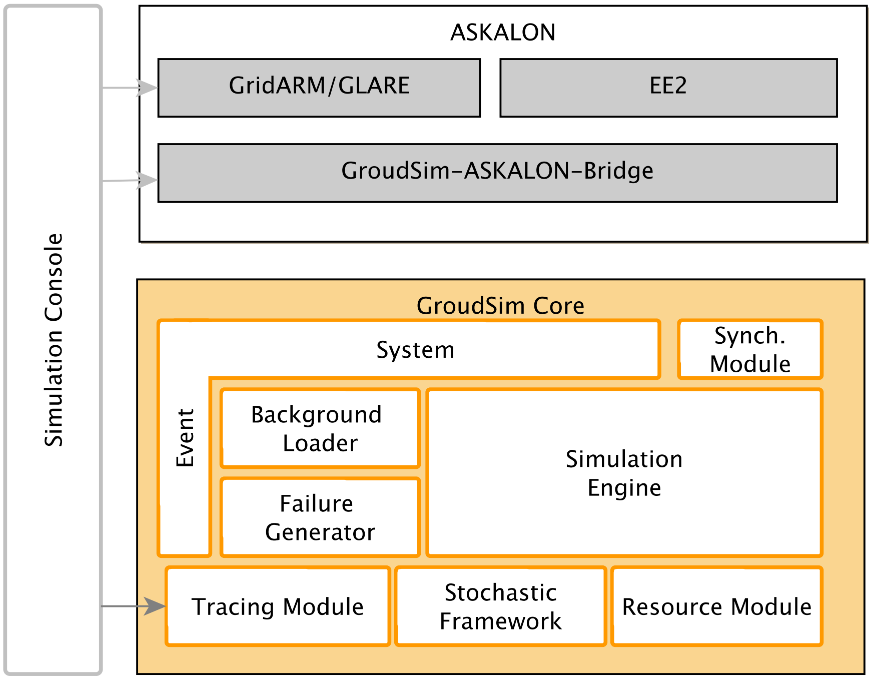

Figure 2. The architecture of GroudSim

Failure generator which handles the failure rates for jobs, file transfers and resources following stochastic distributions;

Stochastic framework that offers different stochastic distribution functions, which can be used for calculating queuing times, submission times, execution times, failure rates or background loads;

Tracing module which is used to store the simulated execution events to a file for analysis or debugging.

In addition to these services, the GroudSim enabled execution and simulation service provides a GUI that allows easy setup of the Grid and Cloud resources by showing statistical charts of the simulated tasks and file transfers. All this information can also be collected in the performance database or setup via configuration files, if preferred.

3.3. DISSECT-CF

As we plan to support future research in the ASKALON workflow enactment, we aimed at increasing the capabilities of GroudSim with the least effort. Unfortunately, GroudSim's lack of internal IaaS behavioural knowledge reduces the number of future use cases that could be supported in the ASKALON-GroudSim system. Therefore, we have analysed several simulators that could act as the foundation for GroudSim and offer insights about the internals of IaaSs. Since GroudSim's focus was primarily on performance and efficiency, we selected the DISSECT-CF simulator to complement its functionality that has a good performance, while having similar internal concepts of time, events, and infrastructure

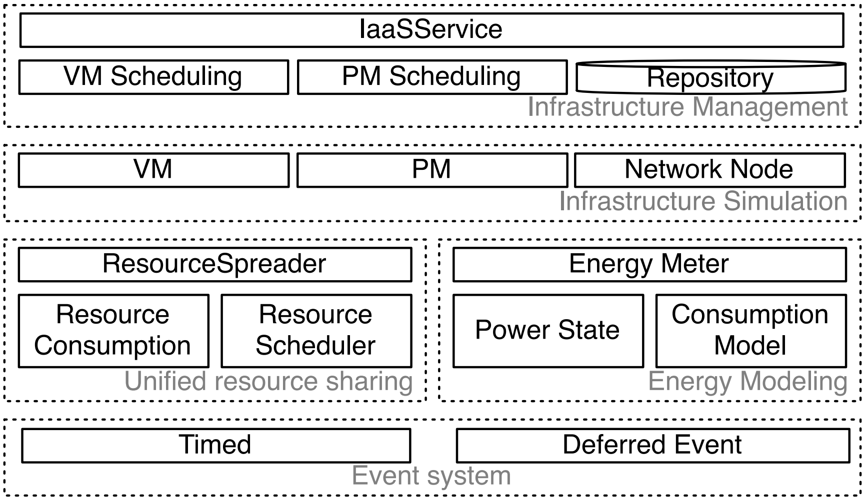

Figure 3. The architecture of DISSECT-CF

DISSECT-CF [11] is a compact, highly customisable open source Cloud simulator with special focus on the internal organisation and behaviour of IaaS systems. Figure 3 presents the architecture of the currently available‡ 0.9.5 version. The figure groups the major components into subsystems marked by dashed lines. Each subsystem is implemented as independently from the others as possible. There are five major subsystems, each responsible for a particular aspect of the internal IaaS functionality: (i) *event system* for a unified time reference; (ii) *unified resource sharing* to resolve low level resource bottleneck situations; (iii) *energy modelling* for the analysis of energy usage patterns of individual resources (e.g., network links, CPUs) or their aggregations; (iv) *infrastructure simulation* to model physical and Virtual Machines (VMs) as well as networked entities; and finally (v) *infrastructure management* to provide a real-life Cloud API and encapsulate Cloud-level scheduling.

After the simulators are integrated, the new ASKALON workflow enactors can perform better by utilising more information than the previously available job run-times and VM execution prices. Thanks to DISSECT-CF, the new enactors will be capable to use VM instantiation timings, job/VM or even workflow level energy consumption details and a more precise network and CPU process model. On the other hand, DISSECT-CF, if used through GroudSim, will immediately gain Cloud pricing capabilities and the possibility to involve hybrid workloads by utilising both Clouds and Grids in a single simulation. The following section details how the integration of the two simulators enables these new functionalities.

4. ASKALON, DISSECT-CF, AND GROUDSIM INTEGRATION

Throughout the integration of the three simulators, we aimed at maintaining the API compatibility of GroudSim, ensuring that the past work on GAB does not need to be repeated. We investigated the APIs of both GroudSim and DISSECT-CF and we analysed the bridging functionalities

‡https://github.com/kecskemeti/dissect-cf

DOI: 10.1002/cpe

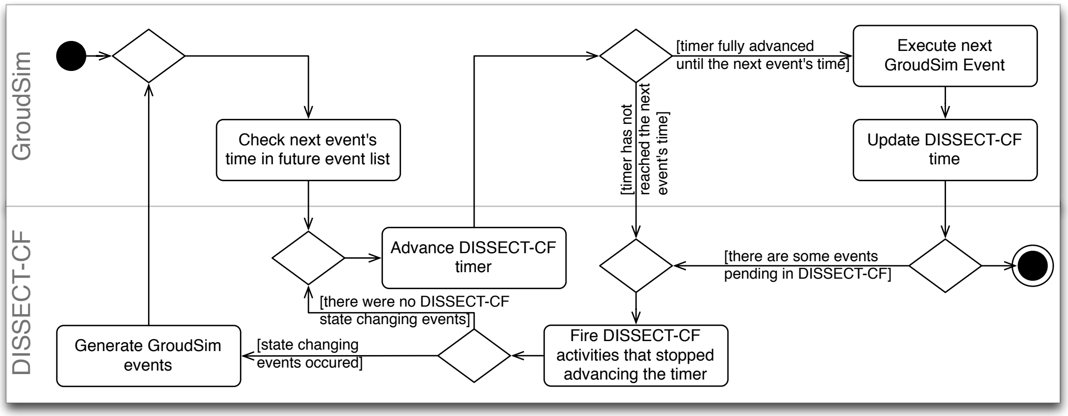

Figure 4. Interaction between GroudSim and DISSECT-CF within the main event processing loop of GroudSim's simulation engine (the diagram only captures the processing of a single GroudSim event within the loop).

needed to cross the simulators' boundaries. According to our analysis, there are three major areas where the simulators have significantly different but relevant APIs for our goals: (i) the event systems with different event types, event firing mechanisms, clock maintenance techniques; (ii) the Cloud representations with conceptual disagreements on data centre organisation, VM and job management mechanisms; and finally (iii) network construction, utilisation, sharing and organisation. The rest of this section discusses how we closed the gaps amongst these areas to allow a seamless transition from GroudSim level simulation to the abstraction used in DISSECT-CF.

4.1. *Event systems*

First of all, we chose to make GroudSim the master simulator. As a result, there should be no events in DISSECT-CF unless there was a preceding GroudSim event that caused a series of DISSECT-CF ones. Second, if some activity happens in DISSECT-CF that has an equivalent one in GroudSim, it must be ensured that DISSECT-CF-level events are never directly sent to the GroudSim user. Instead, they must set off an equivalent GroudSim event (see Figure 4's GroudSim event generation activity). This technique ensures that simulations utilising GroudSim features never need to be aware of the internals of the DISSECT-CF based activities, and increases the performance of the integrated simulators by reducing the number of events that must go through GroudSim.

Now that we saw how events occur across simulation boundaries, let us focus our attention on the way the timing of these events is simultaneously managed in both simulators. To keep the two simulators synchronised, we extended the simulation engine of GroudSim, as depicted in Figure 4. This extension alters the simulators' future event list processing and inside its event loop it always ensures that at any given time instance neither DISSECT-CF nor GroudSim has events, which should have happened already according to the maintained time in the other simulator. The time of GroudSim is kept synchronised with DISSECT-CF by ensuring that only GroudSim's simulation engine controls the time of the underlying DISSECT-CF simulation (see the time advancement and update activities). The extension also handles situations when events in one of the simulators cause events in the other one, as shown by the last conditional activity on the GroudSim side in the figure. This is especially important as DISSECT-CF has two kinds of events: time and state-dependent

DOI: 10.1002/cpe

ones. The *time-dependent events* are placed in the event queue of the `Timed` class, while the *state-dependent events* are fired by the entities that have had their states observed. In GroudSim, these two kinds of events are linked with the technique of event references, and during creation every event has its predetermined occurrence time. However, to reduce the synchronisation overhead, we chose not to create event references in sync with GroudSim, as the occurrence times are not yet available for state-dependent events. Instead, when a state-dependent event occurs, we request GroudSim to insert a new event into its queue for immediate execution, as displayed in the end of the DISSECT-CF activities in Figure 4.

4.2. Cloud representation

Originally, the two simulators approached infrastructure Clouds in a conceptually different way. While GroudSim focused mostly on the black-box Cloud model, DISSECT-CF offered insights on the internals of IaaSs. The black-box model allowed GroudSim to abstract away activities like VM creation details, VM placement, and physical machine state scheduling, which ensured fast and efficient evaluation of Cloud-related workloads. Unfortunately, the black-box model cannot be successfully applied to Cloud infrastructures with limited resource capabilities such as private or academic Clouds, because the abstracted activities can make a significant differences to the outcomes of VM operations. Thus, we decided to keep the API of GroudSim but dropped the black-box model, and ensured that DISSECT-CF simulates the previously abstracted functionalities. Although this addition introduces some performance penalties, we chose DISSECT-CF because it showed better performer than other simulators with similar features. [11]

4.2.1. Cloud infrastructure management. Because of its black-box approach, GroudSim defines Clouds using two properties: the number of processing cores and the set of VM instance types one can create. On the other hand, in DISSECT-CF one can define the kinds and the amounts of physical machines, energy consumption properties, and custom VM and physical machine schedulers. The integrated version introduces more flexibility to GroudSim's Cloud representation in the following two aspects: (i) limited customisability restricted to the number and kind of physical machines; and (ii) extended customisability enabling better energy awareness through customisable consumption properties and physical machine schedulers. Unfortunately, even with the extended approach the customisation of VM schedulers is not entirely possible because GroudSim expects Clouds to reject VM instance requests that cannot be served in the current state of the simulated IaaS. This behaviour is similar to what we currently observe in commercial and academic Cloud systems (like OpenNebula).

As far as the limited customisability is concerned, the GroudSim user is not expected to know that in the background there is another simulator responsible for the internals of Cloud infrastructures. In this case, we expect that the users first define what kind of instances they need from a particular Cloud. Our approach then determines the maximum number of CPUs, the top performance (in terms of MIPS per core), and the highest amount of memory needed by any of the user-defined instances, used for the definition of the physical machine template which is the foundation of the DISSECT-CF Cloud infrastructure. In DISSECT-CF, we create as many of these kinds of physical machines as the amount of CPU cores asked by the user for the particular Cloud during its construction. The created physical machines are connected via a Cloud-level network. The internal DISSECT-CF

DOI: 10.1002/cpe

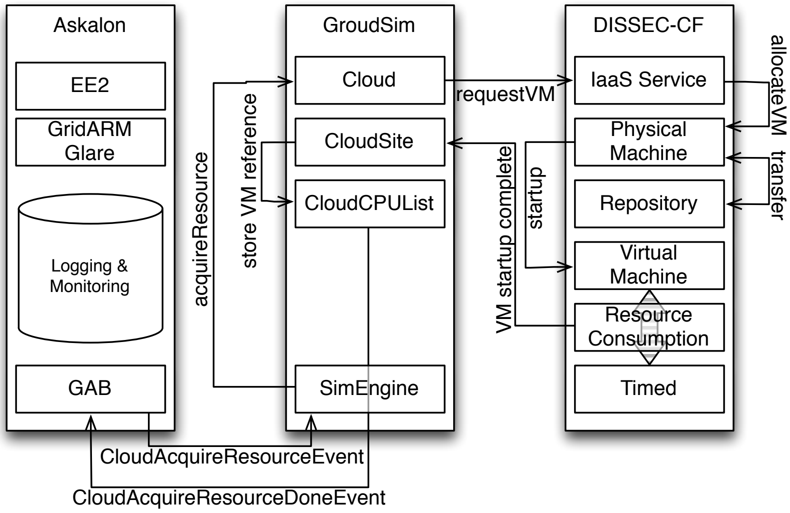

Figure 5. Extended VM instantiation procedure

Cloud representation also simulate a repository to store a single kind of virtual appliance from which all the VMs can be derived. The physical machines are controlled by a physical machine scheduler that keeps them always available. As it can be seen, this infrastructure is rather limited and, as a result, it also limits the possible evaluation scenarios supported by the integrated simulators.

To remove some of these limitations, the simulator allows loading of the internal Cloud properties via a file. In this file, users can define the topology of the physical machines, custom power profiles, and physical machine schedulers. These alterations enable network and energy-aware workflow enactment, but demand user knowledge on the creation of the Cloud description file that DISSECT-CF can process with its `CloudLoader`. Fortunately, the Cloud description file allows keeping the GroudSim APIs unchanged and altering the IaaS behaviour from one simulation run to another.

4.2.2. VM representations. DISSECT-CF allows flexible and continuous resource constraint control during its VM instance creation mechanism (i.e., users can ask for arbitrary memory and processing capabilities), similar to the behaviour of several academic Cloud wares. Unfortunately, similar to Amazon EC2, the instance type system of GroudSim significantly limits the possible kinds of VM instances the users can create. To keep the Cloud concept of GroudSim in the integrated simulation, we have limited the continuous resource constraint space of DISSECT-CF to the instance types of GroudSim.

To uniformly handle Grids and Clouds, GroudSim considers a single VM as a "`CloudSite`". Therefore, such resources are scheduled by the operating system-level schedulers instead of local resource management systems applied in Grid systems. As depicted in Figure 5, `CloudSites` in GroudSim are requested with an instance type, which is then forwarded to DISSECT-CF that schedules the VM to the most suitable physical machine. If the current physical machines in the Cloud are overloaded and cannot serve the requested instance type, the VM scheduler will mark the requested VM as non-servable, allowing DISSECT-CF to fail the `CloudSite` acquisition process in GroudSim. If the VM request can be allocated to a physical machine, the VM is instantiated on

DOI: 10.1002/cpe

it by simulating the transfer of its virtual appliance to the necessary storage element followed by its startup procedures. Finally, the GroudSim user is notified about the creation of the new VM.

It is important to mention that in GroudSim one cannot provide any relevant information to differentiate the planned function of newly created VMs. As a result, GroudSim always instantiates DISSECT-CF VMs with the same virtual appliance. As appliance size is a significant factor in VM creation time, this loss of differentiation between virtual appliances significantly reduces the variance of VM creation times. Therefore, even in DISSEC-CF enhanced GroudSim simulations, the variance of a particular appliance's transfer can be affected only by network activities like transfers between VMs or significant VM creation bursts. We plan to extend GAB so that it is able to forward the expected functionality of a future VM by sending the properties of the applications planned to be run on the VM upon creation.

4.2.3. Job scheduling. Since GroudSim did no mapping between physical and VMs, there was no chance to observe several phenomena that occurs in under-provisioned Clouds. `CloudSites` processed jobs independently from others in the particular Cloud infrastructure, although they could share resources in the background. This sharing reduces the accuracy of GroudSim in scenarios involving heavy Cloud usage. Jobs in GroudSim are also restricted to using single CPU cores. In Grid sites, this restriction is further extended so that one CPU is not allowed to have multiple jobs. GroudSim removes this restriction for Clouds and, as a result, it allows the simulation of simple VM level resource bottlenecks. Unfortunately, these bottleneck situations are less frequent in Clouds, especially with job models that cannot be suspended if needed (e.g., because of changing application characteristics or job migration across VMs). These issues hinder the evaluation of advanced scheduling and workflow enactment techniques in ASKALON. With its unified resource sharing mechanism, DISSECT-CF offers a widely applicable resource scheduling technique that can efficiently and more accurately manage resource bottleneck situations.

After the integration, GroudSim simulates a job in two phases. First, it manages the job's lifecycle until running on a `CloudSite`. In this case, the selected `CloudSite` injects a new CPU-level resource consumption into the VM representing the site in DISSECT-CF. In the second phase of job execution, DISSECT-CF applies its resource sharing technique that automatically considers both the physical machine's load hosting the VM, as well as the currently processed jobs in the VM. While simulating a Cloud infrastructure with a VM scheduler that allows under-provisioned VMs, the jobs will automatically experience performance drops. Moreover, the integrated simulators allow jobs to have limited performance for some periods of time and cancellation-free migration across other DISSECT-CF simulated VMs.

4.3. Networking

GroudSim offers customisable network links among both Grid- and Cloud sites. These links are connected to GroudSim's central bus representing the Internet (see Figure 6). `FileTransfers` between two sites are passed through two network links and the Internet. When bandwidth is utilised by multiple file transfers, the share of each transfer is estimated based on the network link with the smallest bandwidth. This estimate, however, often leads to unused network capacities and highly inaccurate network bandwidth utilisation compared to real life or packet level simulator results.

DOI: 10.1002/cpe

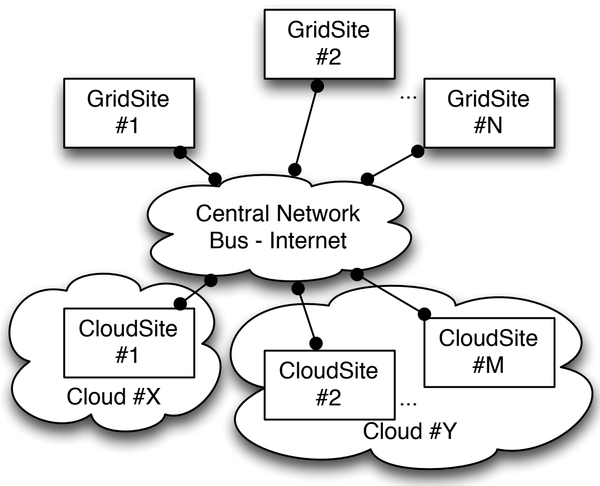

Figure 6. Original GroudSim network topology

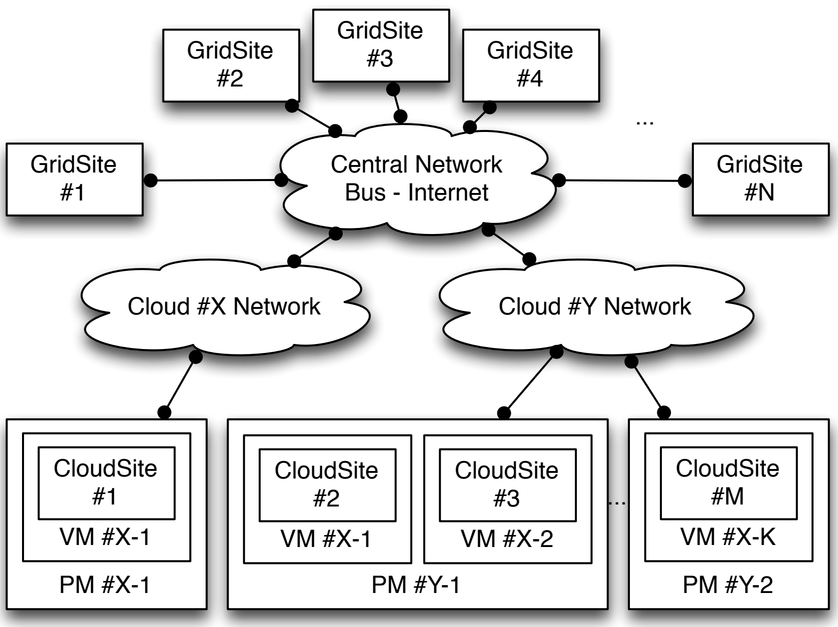

Figure 7. Integrated GroudSim-DISSECT-CF networking

Although, DISSECT-CF still offers a simplified network model, it can model GroudSim's central bus topology without any modifications. Moreover, thanks to its unified resource sharing model, it can immediately offer a solution to network resource bottleneck resolution. Clouds loaded with extended customisability can even limit their overall connections to GroudSim's central bus as a whole. As a result, DISSECT-CF extended simulations can organise GroudSim's Clouds and Grids into a hierarchical network (see Figure 7) where new workflow enactment techniques could investigate the effects of alternative Cloud deployment layouts on network transfers, latencies and VM instantiation times.

Prepared using cpeauth.cls
DOI: 10.1002/cpe

5. EXPERIMENTS

In this section we will present evaluation experiments based on simulation of real world workloads using the original GroudSim and the integrated version to show that the added functionality did not result in significant simulation performance degradation. Then, we highlight new opportunities in the workflow system brought by the new DISSECT-CF integrated features that result in an improved ASKALON environment. Finally, we discuss how the composite ASKALON-GroudSim-DISSECT-CF system scales during the simulated execution of a workflow.

5.1. Scalability

In the first evaluation phase, we aimed at determining the overheads introduced into GroudSim because of the additional features available through DISSECT-CF. We expected a performance drop because of the cross-simulator time synchronisation and the more detailed infrastructure simulation techniques. In order to evaluate the properties of the integration, we chose several simulation scenarios: (i) realistic background loads using GroudSim's background loader and the Grid Workload Archives (GWA); (ii) improved networking capabilities via simultaneous network transfers; and (iii) simultaneous VM instantiation performance for large-scale environments. In all three cases, we performed the evaluation in three phases: (i) implementing the intended simulation in both new and old versions of GroudSim; (ii) comparing the simulation results of the two versions, and (iii) comparing the simulation times. We repeated each simulation setup and collected the simulation time t_{set} until the standard deviation s_N between two consecutive simulations becomes stable below 1%:

$$\frac{s_N(t_{set}) - s_{N+1}(t_{set})}{s_N(t_{set})} < 0.01, \text{ where } N \geq 2 \tag{1}$$

This produced in practice between $7 - 22$ repetitions from which we reported the medial as the t_{set} value.

GWA traces. We selected the GWA real-life scientific workloads for which GroudSim already has a Background Loader (see Fig. 2). Unfortunately, the GWA focuses on Grid workloads that are not entirely suitable in a Cloud context. Although there are several workload traces already available in the Cloud computing community, they are mostly limited to VM management operations and they rarely include VM-to-task allocation information, an important aspect exploited in GroudSim. Moreover, these newer traces do not include enough information on user activities to evaluate the scalability penalties introduced in our integrated simulators. Thus, instead of using such Cloud-specific workloads, we extended the GWA traces to include VM management and VM-to-task allocation as described in the following paragraph.

First of all, we interpreted job submissions in the GWA traces as VM requests. We determined the number and the kind of VMs requested by the number of processors required by the particular job to be run on the VMs. For example, if the Cloud provides VM types with 1, 2, 4, 8, 16, 32 and 64 cores and the job requires 1024 processors, we would request 16 VMs with 64 processors. If the simulated Cloud cannot serve the requested VMs at once, we applied a simple policy that delays the VM requests until of the previous jobs completed. Finally, upon receiving notification from the simulator that the VMs for a specific job are ready, we allocated the job to the newly prepared

Prepared using cpeauth.cls
DOI: 10.1002/cpe

Figure 8. Scalability analysis of GroudSim and DISSECT-CF by simulating job-trace-files

Figure 9. Relative performance drop of the extended simulator compared to its original form

VMs in parallel fragments so that the VMs are completely filled. In our previous example, the 1024 processor job would be split to use all 16 VMs in parallel.

For this experiment, we prepared a Cloud infrastructure with a resource pool of 50 physical machines, each equipped with 64 cores, 128 gigabytes of memory and 5 terabytes of disk, and executed the first C jobs (ranging from 100 to 200.000) from the Auver Grid and from the Grid5000 traces using both the original and the extended GroudSim simulator.

The performance results in Figure 8a small increase in simulation time for the integrated GroudSim – DISSECT-CF simulator that does not harm the scalability with the total number of jobs significantly. By comparing this overhead to the improved, more versatile and feature-rich simulation, we concluded that this slightly longer simulation time is reasonable.

In terms of relative performance degradation compared to the original simulator, Figure 9 shows that the Auver Grid trace increases the simulation time by 79% for most of the scenarios.

DOI: 10.1002/cpe

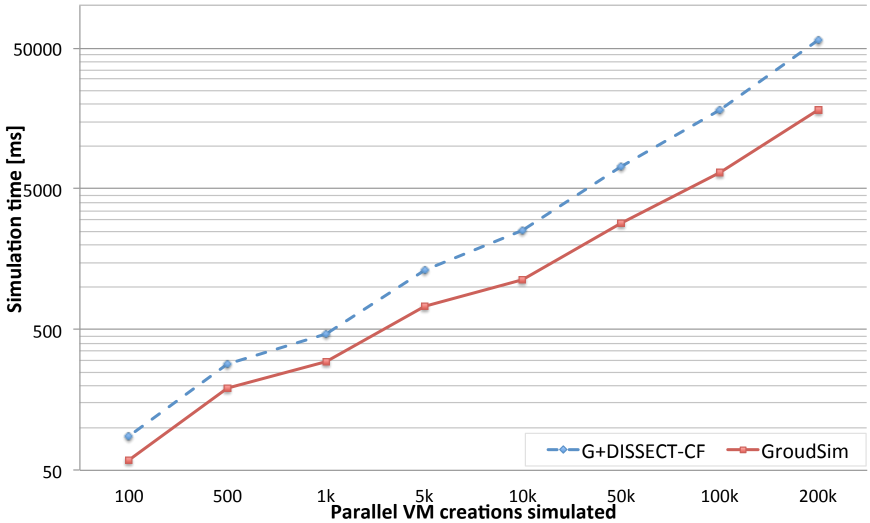

Figure 10. Scalability analysis of GroudSim and DISSECT-CF by requesting VMs in parallel
.

Grid5000 was less affected by the integration and only had an additional execution time of 48% when simulating 100 tasks. For larger simulations, we reached again a value of around 80%. The figure also shows that for smaller scale experiments, the relative performance of the simulators significantly varies due to the internal Java Virtual Machine behaviour. On the other hand, after reaching around 10.000 simulated jobs, the overhead is ranging between 62% – 101% depending of the content of the traces. According to our experiences, it never reaches more then 120% even with different traces than the ones we presented here.

VM creation. As the simulator is intended for evaluation of practical scenarios, we investigated a typical use case in a Cloud setting where the virtual infrastructure of a Software-as-a-Service (SaaS) provider needs to scale-up rapidly. In this case, the IaaS Cloud operator is requested to deliver a certain amount of new VMs to the SaaS provider in a timely fashion. For this scenario, we simulated a large enough Cloud infrastructure to host the requested VMs of the SaaS provider. Next, we prepared several traces (in the GWA format) that simultaneously requested $J \in \{10^2, 5 \cdot 10^2, 10^3, 5 \cdot 10^3, 10^4, 5 \cdot 10^4, 10^5, 2 \cdot 10^5\}$ single processor jobs, translated into small compute instance requests to our simulated Cloud. After the simulators executed all jobs in the traces, each of them in a separate small compute Cloud instances, we analysed the total simulation time and how the individual jobs performed.

Figure 10 shows that the larger the parallelism, the higher is the scalability impact of DISSECT-CF to GroudSim with respect to the parallel VM creations, the maximum being over 3.1 times. This is caused by the extra simulation efforts made by DISSECT-CF for VM startup, network/disk transfers during VM image delivery, VM scheduling, and so on. The higher performance degradations (i.e., over 2.2 times for $J \geq 10^4$) are mostly due to the Cloud setup in the integrated simulator due to the single VM image store that becomes a bottleneck when more VMs are requested in parallel. Fortunately, the performance degradation is a less concern in smaller Cloud setups with a few thousands of computing nodes, or for less parallel situations (i.e., request rates lower than

DOI: 10.1002/cpe

a few hundreds), which is mostly the target area of GroudSim. In the future, we plan to improve the automated Cloud creation process to better match the properties of the underlying DISSECT-CF simulation by creating multiple VM image stores, in parallel with further improvements in DISSECT-CF under such level of parallelism.

Networking. For this experiment, we set up two hosts in the simulators and connected them with a gigabit Ethernet network and simultaneously started $T \in \{1, 2, 5, 10, 20, 50, 100, 200, 500, 1000, 2000, 5000\}$ transfers. We selected the numbers for simultaneous transfers to range between average (e.g., $T = 1 - 10$) Cloud and more HPC-like systems (e.g., $T > 50$). For each network transfer, we selected the source and the target hosts in a round-robin fashion. We simulated each transfer to be $B \in \{1, 10^2, 10^4, 10^6, 10^8\}$ bytes in size. Similarly to the number of simultaneous transfers, we selected the transfer size to allow the simulation of RPC (e.g., $B < 10^4$) for transferring complete VM images ($B \geq 10^8$). Our experimental setup has two parameters: T and B and run the experiments with all possible combinations of $T \times B$. We compared the simulated transfer times of each transfer in both integrated and old stand-alone GroudSim simulator. We found out that the median difference between the two is negligible, smaller than the time resolution of the simulator. In contrast, the average is slightly over 31 milliseconds (1.2%) resulting from the different number representations in the two simulations.

Next, we switched to the scaling experiment of this network setup by running each experimental setup at least 10 times according to the Equation 1. Compared to the other scalability measurements, Figure 11 shows a different behavior. First, we see that for transfers with little concurrency (i.e., $T \leq 5$), the performance of the combined simulators is lower like in the other performance evaluation scenarios around 2.3 times. In contrast, the highly concurrent scenarios show that by letting DISSECT-CF manage the entire network stack underneath GroudSim brings significant advantages. For the transfers in the HPC range ($T > 50$), the average performance improvement is 41 times. The performance improvements are mostly the result of the unified resource sharing foundation of DISSECT-CF, which is especially tuned for highly parallel resource sharing scenarios.

5.2. ASKALON research opportunities.

DISSECT-CF brings new features into the ASKALON ecosystem that will allow scientists, application developers and the ASKALON team to extend and improve their research in multiple areas and directions. We introduce in the following the new research areas available to ASKALON as a result of the new integrated simulations, while in Section 6 discuss additional research areas planned to be covered in the future.

Network use. GroudSim was originally developed with little focus on network functionality the focus of all workflow applications used in ASKALON was on the computational part. In these workflows, file dependencies comprised only a marginal amount of data that had to be transferred between resources. With integration of DISSECT-CF, the network model of GroudSim was replaced by a more accurate one that allows more precise simulation of data-intensive applications. As a result, scheduling techniques that consider data movement can exploit the more accurate file transfer predictions to improve the workflows execution times.

DOI: 10.1002/cpe

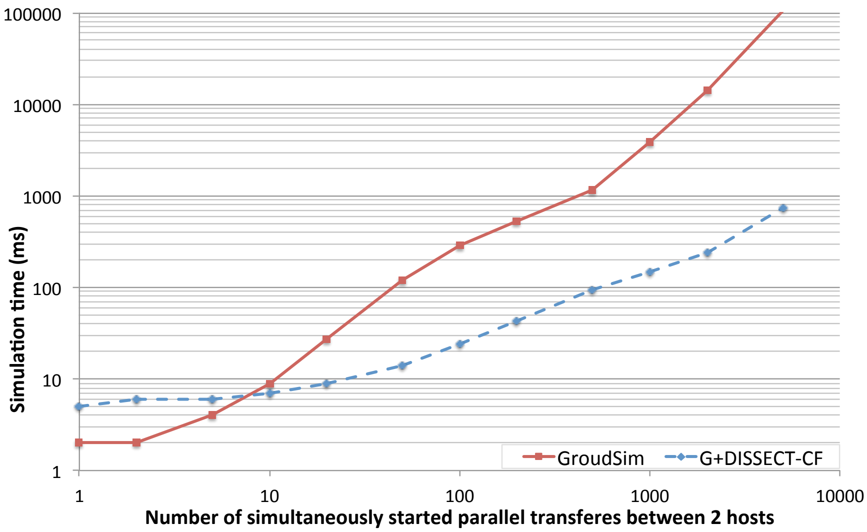

Figure 11. Scalability analysis of GroudSim and DISSECT-CF for network transfers.

Data centre configurations. DISSECT-CF allows to specify the characteristics of data centres in an easily exchangeable configuration file. Utilising this mechanism gives us the opportunity to evaluate the kind of data centre that best fits a specific workflow application. This feature not previously possible with GroudSim, as the hardware model of data centres was not existing. In the near future, we will aim at determining the influence of the data centre configuration on workflow applications and their schedule by showing how important is to simulate Clouds not only as black boxes, but to consider their internal hardware characteristics too.

With the ongoing development of DISSECT-CF, the supported research directions will be further extended. ASKALON users and developers will directly benefit from each new feature developed in DISSECT-CF and will allow scientists develop new methods, algorithms and solutions to Cloud and workflow management-related problems.

5.3. Workflow simulation.

In this section, we demonstrate the use of ASKALON to simulate the execution of scientific workflow applications on IaaS Cloud resources using the integrated simulators described in the previews sections. We preformed a set of experiments based on real execution traces to illustrate the use of GroudSim for simulating runs would otherwise take weeks and exhibit high operational costs on commercial Clouds.

The integration of DISSECT-CF into GroudSim provided access to the new features added and higher simulation accuracy to the ASKALON environment. In ASKALON, workflows can be executed in real environments, as well as in simulated environments allowing the evaluation of new execution scenarios without needing to actually pay for the underlying infrastructure. Currently, the simulated workflows can run on dynamic Cloud resources leased from a IaaS provider, as well as on more static Grid resources similar to academic infrastructures as the Austrian Grid. To allow faster evaluation of new workflow execution techniques and optimizations, simulation is widely used

Prepared using cpeauth.cls
DOI: 10.1002/cpe

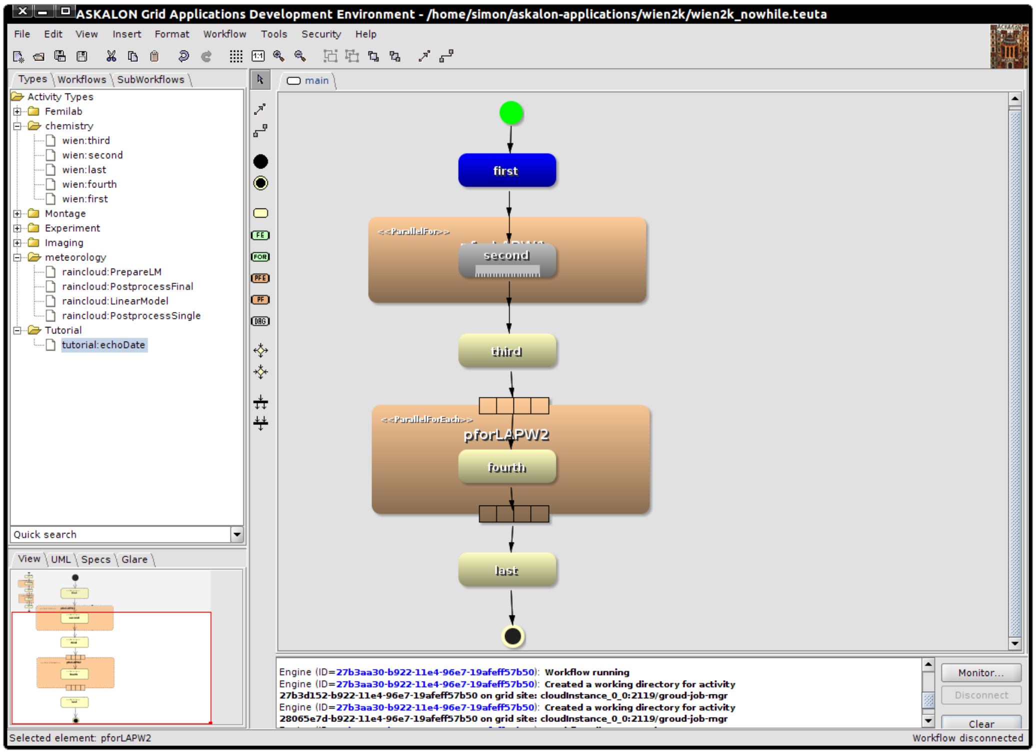

Figure 12. Workflow creation and execution GUI (Wien2k [27]).

in workflow research. It allows important observations such as enactment overheads, scheduling inaccuracies, resource utilisation, execution cost, or energy consumption to be collected.

Figure 12 shows a screenshot of the main ASKALON GUI used for developing workflows and submitting them for execution (i.e. the Wien2k workflow explained in Section 5.4). In the top-left tree, a list of available applications is shown to the user that can be drag-and-dropped in the main area on the right side. Activities are connected with control flow edges from the initial node towards the final node of the workflow. The bottom-left part shows the services the GUI is currently connected to, and log messages of the ongoing execution.

In most environments, the scientists first craft their ideas on paper, then implement them in a simulator and finally, if the simulation results show an improvement over the existing technologies, they implement them in a real system. In case of ASKALON, the second and the third step is combined, as the initial implementation for the simulated evaluation is already done in the real system. Switching from simulation to execution requires changing a single flag and all resource requests, file transfers or job submissions are either sent to GroudSim or are executed on a real environment.

Figure 13 shows the ASKALON's monitoring tool used to visualise the execution in a Gantt Chart fashion. The left side gives a tree representation of the unrolled workflow, allowing the user to select the workflow regions to be displayed on the right side

The GroudSim-enabled execution service additionally provides the user with a GUI that allows to simply create, store or save a infrastructure configurations (see Figure 14). Users can set up infrastructures with a mixture of Cloud and Grid resources (ASKALON supports execution

DOI: 10.1002/cpe

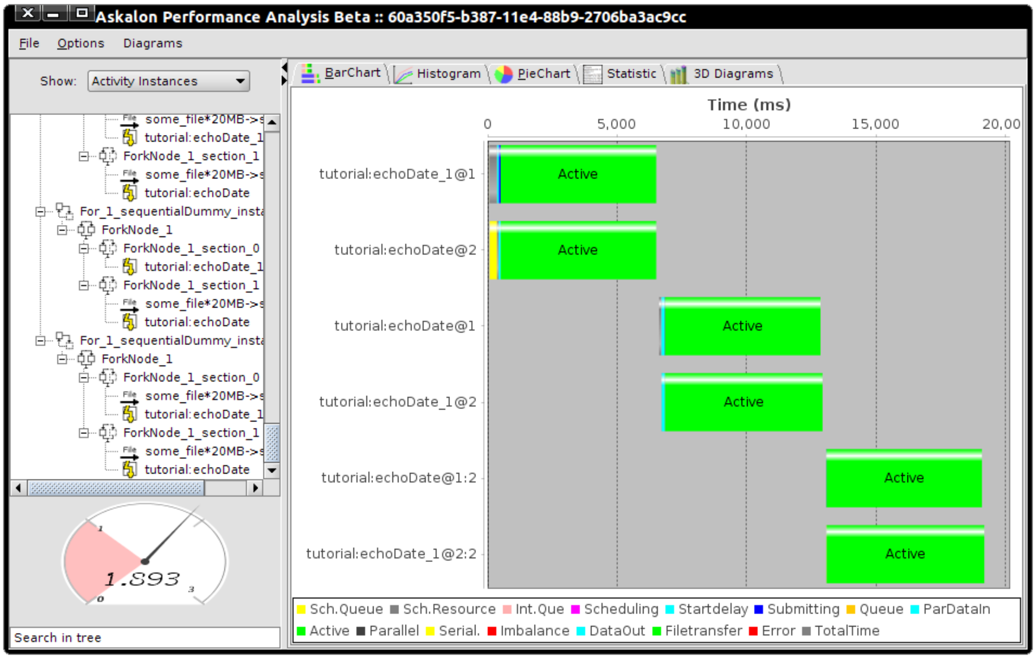

Figure 13. ASKALON workflow monitoring and performance tool.

on such hybrid hardware environments). The GUI displays simple statistics like the number of simulated jobs or file transfers and provides means to start or stop Cloud instances, define resource requirements of tasks, or give custom file sizes if needed. All settings are accessible via the GUI and can also be loaded using configuration files to allow massive simulation runs that can be run in a automatic fashion. We developed additional tools to manage and monitor the simulation because, although the ASKALON monitoring tools work with the simulated workflow executions, if hundreds workflows are executed it is more user-friendly to see the complete execution information while ASKALON is oriented to visualize them per workflow execution.

Simulation experiments are especially important for use cases which are hard to (re)produce, such as resources failures, network interruptions, spot instances terminations, and similar. These events might occur any time during a workflow execution and therefore a workflow system should be able to deal with such faults. As physical access to networks are not always granted, it may be impossible to physically take one region of a Cloud provider (i.e. Amazon EC2) offline without accessing the network configurations beyond one's control. Simulated executions allow such behaviour reproductions.

5.4. Wien2k.

Wien2k [27] is a material science workflow for performing electronic structure calculations of solids using density functional theory based on the full-potential (linearized) augmented plane-wave ((L)APW) and local orbital (lo) method. The Wien2k workflow contains two parallel sections of size x, with sequential synchronization activities in between. Figure 15 shows the complete unrolled workflow, modelled in a more compact form in the ASKALON GUI as shown in Figure 12. The total number of activities in a Wien2k workflow is: $N_{wien2k} = 2 \cdot x + 3$, where x is the number of parallel activities, also called *k-points*. We executed the simulations on a desktop computer with

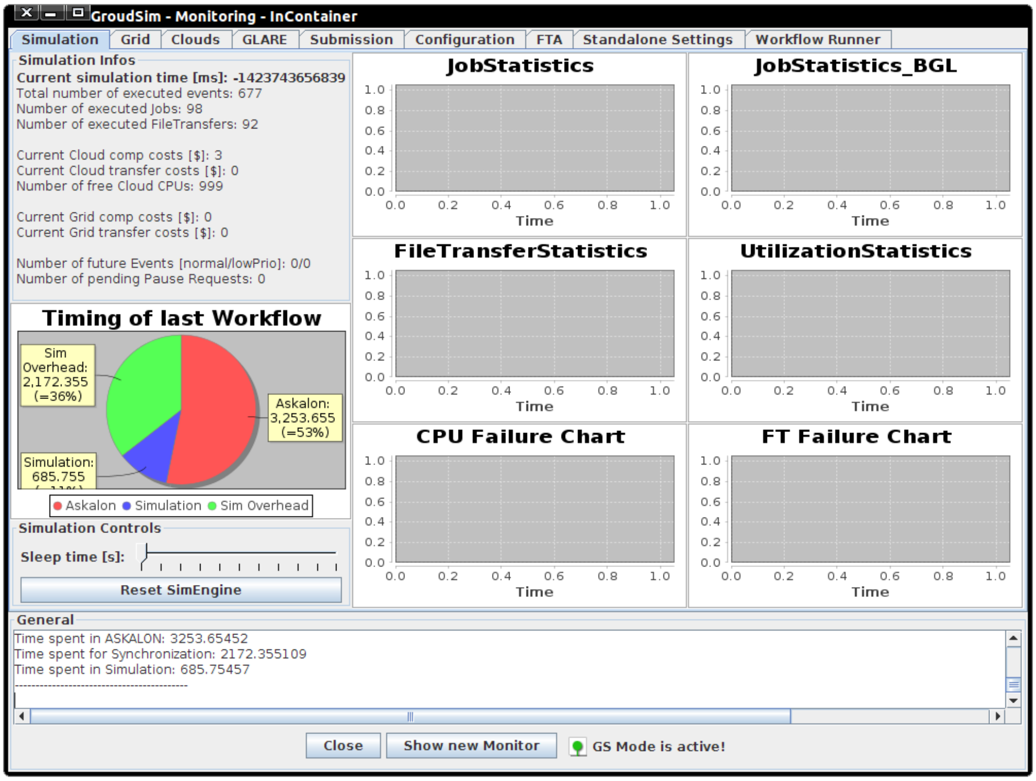

Figure 14. GroudSim monitoring and configuration GUI in ASKALON simulation mode.

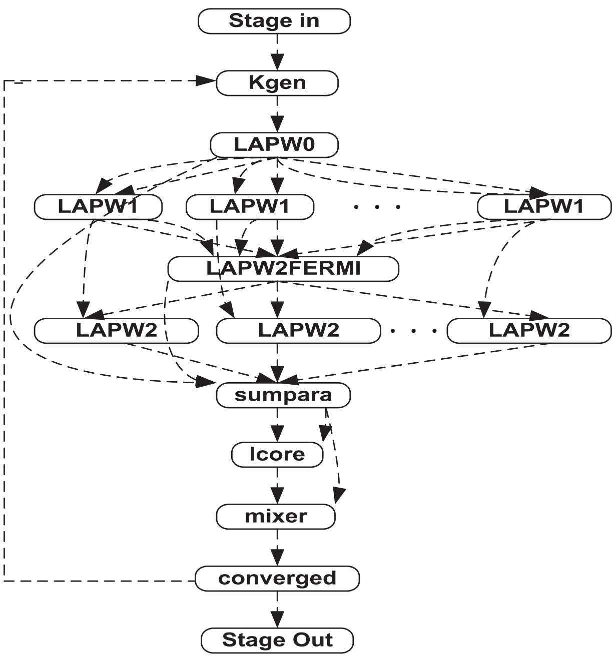

Figure 15. The scientific workflow application Wien2k.

an Intel i7-2600k processor and 8 gigabytes of memory running the Ubuntu operating system. We

DOI: 10.1002/cpe

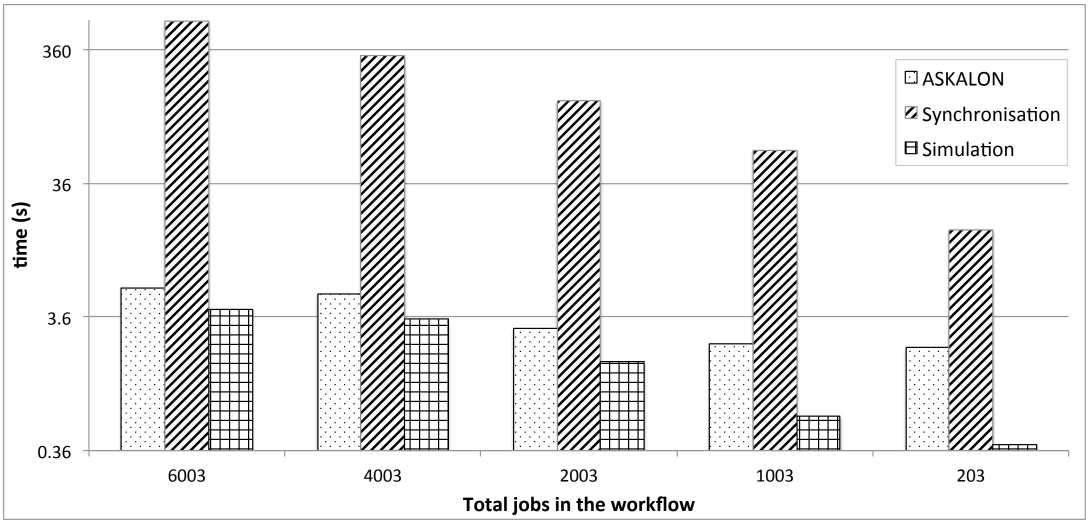

Figure 16. Wien2k simulation times for $x = 100, 500, 1000, 2000, 3000$.

executed all components of the simulations on the same host as a standard use case of developers running simulations directly on their workstations.

Figure 16 shows that simulating a workflow with 100 k-points (resulting in 203 tasks and 1418 file transfers) takes 18.7 seconds in average, while workflows with 3000 k-points (6003 tasks and 42018 file transfers) take in average 10 minutes. The total runtime is split in 3 parts: (i) ASKALON is the time that can be accounted to the execution system, (ii) Simulation is the time spend in simulating executions of tasks, file transfers, VM instantiation and similar, and (iii) is the time spend to synchronize between ASKALON (running in real-time) and the simulators. Parts or ASKALON and Simulation times overlap with the Synchronisation part, as the system was designed for best possible speed using multi threading.

This quite poor performance compared to the GroudSim stand-alone results is explained by the multiple synchronisation points between the simulator and the workflow system. As the figure shows, the simulation's execution time are not significantly influencing the simulated runtime of the ASKALON workflow (because the synchronisation of the ASKALON execution engine and the GAB web services are the biggest bottleneck in our experiments – e.g., for every simulated job the completion is propagated back to the ASKALON GUI which was not an issue in real infrastructures with long runtimes but now it became a serious limitation). Therefore, within the context of ASKALON, we receive the precision and detail improvements of DISSECT-CF practically with no apparent costs for the user.

Taking into account the times only consumed by the simulator, the time for the smallest run with 100 k-points took only 0,4 seconds. The time spend in ASKALON is hard to identify as there is a high overlap with synchronisation times. ASKALONs execution engine is heavily thread based and only a few of them get blocked in the synchronisation parts and have to wait for simulator reactions. 13-16 seconds are covering this synchronisation part and about 1-3 seconds can be addressed to ASKALON only. To improve this performance we plan to work on the GAB to allow bundled event transfer between ASKALON to the simulator to reduce the amounts of changes between simulation and real-time behaviour. Once the GAB allows bundled event transfer the performance degradations

DOI: 10.1002/cpe

Table I. Real world execution cost and time of Wien2k runs from simulated examples.

k-points	tasks	file transfers	cost [$]	runtime [h]
100	203	1418	53,46	127,3
500	1003	7018	320,14	623,77
1000	2003	14018	637,7	1235,1
2000	4003	28018	1273,0	2482,5
3000	6003	42018	1908,2	3692,8

in the integrated GroudSim will be more apparent but we expect that the synchronisation time will still dominate the executions, thus in the future even more features could be added to GroudSim without losing user side performance. Again the integration of DISSECT-CF does show very little influence on the overall performance and does not become the bottleneck of the workflow simulation system.

We collected the cost for the Cloud resources we used within those simulated executions and present those results in table I. For the cost calculation in the simulator we used the informations provided from Amazon EC2: c3.2xlarge with 8 cores and a total of 28 ECU, 15 gigabyte of memory and 2 x 80 SSD hard drives (which are not simulated by the current system) for $0.420 per Hour based on the US East regions cost.

Table I shows that it would take 22 weeks an cost nearly $2.000,0 to run the biggest workflow on an environment like Amazon EC2. Even though execution time might be decreased with more resources in parallel but still the cost would slightly increase with higher resource usage. Comparing this to the simulation time of only 10 minutes and no cost except of the power for the desktop computer let us conclude, that simulation is a important tool for workflow developers. When new optimizations are evaluated and multiple workflow executions are needed to validate the approach, simulation is often the only feasible solution.

6. CONCLUSION AND FUTURE WORKS

When evaluating scientific research, simulation tools are invaluable alternatives to real-world environments. For example in the field of scientific workflow management systems, simulators enable faster, more versatile, deterministic, and reproducible experimentation, including situations not easily reproducible in real-life. Despite their importance, current Cloud workflow simulators lack sufficient support with respect to the underlying virtualised infrastructure, including energy-awareness that is highly demanded in today's data centres. To address this gap, we presented in this paper the integration of a stand-alone DIScrete event baSed Energy Consumption simulaTor for Clouds and Federations (DISSECT-CF) with a mature real-world Cloud workflow management system called ASKALON and its underlying Grid/Cloud simulation environment called GroudSim. We discussed the challenges that appeared as the result of the originally incompatible APIs and functionalities of ASKALON, GroudSim and DISSECT-CF, and presented the required re-engineering and adjustments in three main areas: (i) event system, including event types, firing mechanisms and clock maintenance techniques, (ii) Cloud representation at the level of data centre, VM and job management, and finally (iii) network construction, utilisation, sharing and organisation.

Prepared using cpeauth.cls
DOI: 10.1002/cpe

Our experimental evaluation, conducted on an over 3000 core simulated Cloud infrastructure, demonstrated an improved behaviour of the ASKALON system regarding networking, energy metering, VM instantiation and CPU sharing accuracy while the performance of the integrated simulators never dropped below half of the original GroudSim based simulations. We concluded that despite the improved functionality, the scalability of the simulator did not drop and was in alignment with our past results where we have shown the scaling issues in relation with simulators [7] and evaluated the performance of simulating large skale workflow experiments with the ASKALON environment. We identified optimization possibilities in the following fields: (i) resource utilisation improvements, (ii) power consumption optimisations for workflows and Cloud providers, (iii) network aware workflow scheduling, and (iv) optimising workflow executions depending on data centre configurations.

Future work will target improvements in the GroudSim-ASKALON bridge allowing more information to be shared with the simulators regarding the executed workflows and also allowing ASKALON environment to gather more details about the simulated infrastructures. We will also focus on reducing the performance overheads caused by the duplication of some functionalities in the system (e.g., eventing) allowing GroudSim to concentrate more on the user side behaviour of Clouds and Grids. Finally, we plan to introduce dynamic pricing models to GroudSim by relying on DISSECT-CF's resource utilisation and energy consumption related reports. For the following research directions, that we target, made possible by this additional feature forwarding from DISSECT-CF to the workflow system, we want to give a short overview of the upcoming research areas:

Power consumption. Green IT is getting more important, as power consumption and resulting CO2 emissions are becoming widely known issues to the general public. Workflow schedulers can offer benefits for customers by improving scheduling and resource management through the use of DISSECT-CF provided measurements about the power draw of physical resources. The collected power measurements then allow the optimisation of workflow execution considering not only cost and time but also energy consumption. Although, contemporary Cloud systems lack this metering functionality, enabling research work on the area will increase demands towards providers and prepare novel workflow management systems for times when such features become available from commercial or academic Clouds.

Cloud providers pay special attention to energy consumption reduction as it can directly reduce data centre operating costs. These cost reductions then can either give a competitive pricing advantage to the provider or increase its margins allowing more funds for its activities. Research in the area of VM placement is therefore not only interesting for users but also providers. Data centres could advertise that they apply environment friendly policies and users that want to support power saving would get attracted by such providers similar to renewable energy producers (which manage to sell their energy in most cases even more expensive then regular providers).

Resource usage. The new functionality of DISSECT-CF allows the identification of different physical machines for the instantiated VMs. This does not only allow to invent new methods for IaaS internal resource mapping but also in combination with workflow execution can improve the resource utilisation. In most cases, Cloud providers are seen as black boxes

Prepared using cpeauth.cls
DOI: 10.1002/cpe

where the mapping of VMs to physical resources can only be guessed. With the integration of DISSECT-CF into GroudSim, new possibilities were opened up in ASKALON schedulers and resource managers. Knowing which instances share a physical machine can be used by the scheduler to map tasks with high data dependencies on instances that are close to each other resulting in reduced data transfer times.

New research directions can also be utilised within the IaaS provider. It is now possible to investigate different policies for physical - VM mapping and their influence on performance, power consumption, utilisation and fairness. To investigate those features, internal IaaS scheduling mechanisms need to be changed which would not have been possible in GroudSim before the integration of DISSECT-CF. GroudSim had anonymous resource pools of cores only and did not understand the concept of physical machines.

Additional research in the IaaS area is planed within the H2020 project named EntICE [28] that will focus on multi parameter optimization of Cloud image repositories, including power as an important parameter.

ACKNOWLEDGEMENT

The work presented in this paper has been partially supported by the Austrian Science fund project TRP 237-N23, by the EU under the COST programme Action IC1305, 'Network for Sustainable Ultrascale Computing (NESUS) and by the EU H20 project ENTICE: dEcentralized repositories for traNsparent and efficienT vIrtual maChine opErations 644179.

REFERENCES

1. Taylor IJ, Deelman E, Gannon DB, Shields M. *Scientific Workflows for Grids*. Workflows for e-Science, Springer Verlag, 2007.
2. Armbrust M, Fox A, Griffith R, Joseph AD, Katz R, Konwinski A, Lee G, Patterson D, Rabkin A, Stoica I, *et al.*. A view of cloud computing. *Commun. ACM* Apr 2010; **53**(4):50–58, doi:10.1145/1721654.1721672. URL http://doi.acm.org/10.1145/1721654.1721672.
3. Plankensteiner K, Prodan R, Janetschek M, Montagnat J, Rogers D, Harvey I, Taylor I, Balaskó Á, Kacsuk P. Fine-grain interoperability of scientific workflows in distributed computing infrastructures. *Journal of Grid Computing* 2013; **11**(3):429–455, doi:10.1007/s10723-013-9261-8.
4. Ahmed A, Sabyasachi A. Cloud computing simulators: A detailed survey and future direction. *Advance Computing Conference (IACC), 2014 IEEE International*, 2014; 866–872, doi:10.1109/IAdCC.2014.6779436.
5. Sakellari G, Loukas G. A survey of mathematical models, simulation approaches and testbeds used for research in cloud computing. *Simulation Modelling Practice and Theory* 2013; **39**:92–103.
6. Calheiros RN, Ranjan R, Beloglazov A, De Rose CAF, Buyya R. Cloudsim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms. *Software: Practice and Experience* 2011; **41**(1):23–50, doi:10.1002/spe.995. URL http://dx.doi.org/10.1002/spe.995.
7. Ostermann S, Plankensteiner K, Prodan R. Using a new event-based simulation framework for investigating resource provisioning in clouds. *Scientific Programming* 2011; **19**(2):161–178.
8. Nuñez A, Vázquez-Poletti JL, Caminero AC, Carretero J, Llorente IM. Design of a new cloud computing simulation platform. *Computational Science and Its Applications-ICCSA 2011*. Springer, 2011; 582–593.
9. Ullman JD. Np-complete scheduling problems. *J. Comput. Syst. Sci.* Jun 1975; **10**(3):384–393, doi:10.1016/S0022-0000(75)80008-0. URL http://dx.doi.org/10.1016/S0022-0000(75)80008-0.
10. Ostermann S, Plankensteiner K, Prodan R, Fahringer T, Iosup A. Workflow monitoring and analysis tool for ASKALON. *Grid and Services Evolution*, Barcelona, Spain, 2008; 73–86.

11. Kecskemeti G. DISSECT-CF: a simulator to foster energy-aware scheduling in infrastructure clouds. *Simulation Modelling Practice and Theory* October 2015 *to appear*; :1–31doi:10.1016/j.simpat.2015.05.009. URL http://dx.doi.org/10.1016/j.simpat.2015.05.009.
12. Rogers D, Harvey I, Huu TT, Evans K, Glatard T, Kallel I, Taylor I, Montagnat J, Jones A, Harrison A. Bundle and pool architecture for multi-language, robust, scalable workflow executions Sep 2013; **11**(3):457–480, doi: http://dx.doi.org/10.1007/s10723-013-9267-2. URL http://link.springer.com/article/10.1007/s10723-013-9267-2.
13. Buyya R, Murshed M. Gridsim: A toolkit for the modeling and simulation of distributed resource management and scheduling for grid computing. *Concurrency and computation: practice and experience* 2002; **14**(13-15):1175–1220.
14. Murugesan S. Harnessing green it: Principles and practices. *IT Professional* Jan 2008; **10**(1):24–33, doi:10.1109/MITP.2008.10. URL http://dx.doi.org/10.1109/MITP.2008.10.
15. Chen W, Deelman E. Workflowsim: A toolkit for simulating scientific workflows in distributed environments. *E-Science (e-Science), 2012 IEEE 8th International Conference on*, IEEE, 2012; 1–8.
16. Casanova H, Giersch A, Legrand A, Quinson M, Suter F. Versatile, scalable, and accurate simulation of distributed applications and platforms. *Journal of Parallel and Distributed Computing* 2014; **74**(10):2899 – 2917, doi:http://dx.doi.org/10.1016/j.jpdc.2014.06.008. URL http://www.sciencedirect.com/science/article/pii/S0743731514001105.
17. Ostermann S, Plankensteiner K, Bodner D, Kraler G, Prodan R. Integration of an event-based simulation framework into a scientific workflow execution environment for grids and clouds. *Towards a Service-Based Internet*, *Lecture Notes in Computer Science*, vol. 6994. Springer: Poznan, Poland, 2011; 1–13, doi:10.1007/978-3-642-24755-2_1.
18. Lee K, Paton NW, Sakellariou R, Deelman E, Fernandes AAA, Mehta G. Adaptive workflow processing and execution in pegasus. *Workshops at the Grid and Pervasive Computing Conference, GPC 2008, Kunming, China, May 25-28, 2008*, IEEE Computer Society, 2008; 99–106, doi:10.1109/GPC.WORKSHOPS.2008.30. URL http://doi.ieeecomputersociety.org/10.1109/GPC.WORKSHOPS.2008.30.
19. Wolstencroft K, Haines R, Fellows D, Williams AR, Withers D, Owen S, Soiland-Reyes S, Dunlop I, Nenadic A, Fisher P, *et al.*. The taverna workflow suite: designing and executing workflows of web services on the desktop, web or in the cloud. *Nucleic Acids Research* 2013; **41**(Webserver-Issue):557–561, doi:10.1093/nar/gkt328. URL http://dx.doi.org/10.1093/nar/gkt328.
20. Farkas Z, Kacsuk P. P-GRADE portal: A generic workflow system to support user communities. *Future Generation Comp. Syst.* 2011; **27**(5):454–465, doi:10.1016/j.future.2010.12.001. URL http://dx.doi.org/10.1016/j.future.2010.12.001.
21. von Laszewski G, Foster IT, Gawor J. CoG kits: a bridge between commodity distributed computing and high-performance Grids. *Java Grande Conference*, ACM Press, 2000; 97–106.
22. Barrionuevo JJD, Fard HM, Prodan R. MOHEFT: A multi-objective list-based method for workflow scheduling. *4th IEEE International Conference on Cloud Computing Technology and Science Proceedings, CloudCom 2012, Taipei, Taiwan, December 3-6, 2012*, IEEE Computer Society, 2012; 185–192, doi:10.1109/CloudCom.2012.6427573. URL http://dx.doi.org/10.1109/CloudCom.2012.6427573.
23. Wieczorek M, Prodan R, Fahringer T. Scheduling of scientific workflows in the ASKALON grid environment. *SIGMOD Record* 2005; **34**(3):56–62, doi:10.1145/1084805.1084816. URL http://doi.acm.org/10.1145/1084805.1084816.
24. Braun TD, Siegel HJ, Beck N, Bölöni L, Maheswaran M, Reuther AI, Robertson JP, Theys MD, Yao B, Hensgen DA, *et al.*. A comparison of eleven static heuristics for mapping a class of independent tasks onto heterogeneous distributed computing systems. *J. Parallel Distrib. Comput.* 2001; **61**(6):810–837, doi:10.1006/jpdc.2000.1714. URL http://dx.doi.org/10.1006/jpdc.2000.1714.
25. Sulistio A, Cibej U, Venugopal S, Robic B, Buyya R. A toolkit for modelling and simulating data Grids: an extension to GridSim. *Concurrency and Computation: Practice and Experience* 2008; **20**(13):1591–1609.
26. Iosup A, Li H, Jan M, Anoep S, Dumitrescu C, Wolters L, Epema DHJ. The grid workloads archive. *Future Generation Comp. Syst.* 2008; **24**(7):672–686, doi:10.1016/j.future.2008.02.003. URL http://dx.doi.org/10.1016/j.future.2008.02.003.
27. Schwarz K, Blaha P, Madsen G. Electronic structure calculations of solids using the wien2k package for material sciences. *Computer Physics Communications* 2002; **147**(1):71–76.
28. February 2015. URL http://www.entice-project.eu.