Meteorological Applications utilizing Grid and Cloud Computing

Simon Ostermann and Radu Prodan
Institute of Computer Science,
University of Innsbruck, Innsbruck, Austria
E-mail: simon@dps.uibk.ac.at

Felix Schüller and Georg J. Mayr
Institute of Meteorology and Geophysics,
University of Innsbruck, Innsbruck, Austria
E-mail: felix.schueller@uibk.ac.at

***Abstract*—Three practical meteorological applications with different characteristics are used to highlight the usability of a computer science workflow middleware called ASKALON by allowing easy access to distributed computing for meteorology scientists. Utilizing Cloud and Grid computing, this paper shows use case scenarios fitting a wide range of applications from operational to research studies with real world examples from meteorological research. The paper concludes that distributed computing is easy usable for meteorological problems using the ASKALON system. This powerful tool extends existing high performance computing concepts and allows simple and cost effective access to computing capacity from Grid and Cloud environments. Additional cost perspectives are given, when Cloud computing is cheaper then in house resources: When resources are utilized only a few hours a day.**

I. Introduction and Motivation

Meteorology is a good example for a science that has an ever growing need for computing power, be it for sophisticated numerical models of the atmosphere itself, model chains like e.g. coupled ocean and atmospheric models or the accompanying activities such as visualization or dissemination. In addition to the increased need for computing power, more data are being produced, transferred and stored, which raises the problem complexity and resulting computational requirements.

Starting in the mid 1990s, the concept of Grid computing, in which geographical and institutional boundaries only play a minor role, came to be a powerful tool for scientists. [1] published the first and most cited definition: *A computational Grid is a hardware and software infrastructure that provides dependable, consistent, pervasive, and inexpensive access to high-end computational capabilities.* In the following years the definition changed to viewing the Grid not as a computing paradigm, but as an infrastructure that brings together different resources in order to provide computing support for various applications, emphasizing the social aspect [2], [3]. Grid initiatives mostly focus on raw computing power (*Compute Grids*) or target the storage or exchange of data (*Data Grids*).

Many initiatives in the atmospheric sciences utilize Compute Grids, i.e. climatological applications using a Compute Grid is the Fast Ocean Atmospheric Model (FOAM) [4]. They performed ensemble simulations of a coupled climate model on the Teragrid, a U.S. based Grid project.

Cloud computing is a slightly newer concept than Grid computing formed not from academia but by commercial companies with strong web hosting and web services background. Amazon realized, that the resources needed for peak usage, like Christmas shopping, is unused most of the year and was therefore made available for rental on a hourly basis in the new offer called Amazon Web Services (AWS).

Resources are also pooled, but contrary to Grids usually within one organisational unit or company. Applications range from computational based resource renting, storage services and offer lots of specialized solutions for more advanced systems like map and reduce framework. Fine grain dynamic billing allows to cut ongoing costs or react on capacity needs with short reaction times.

The most important characteristics of Clouds are condensed into one of the most recent definitions by [5]: *Cloud computing is a model for enabling ubiquitous, convenient, on demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction.* One of the few projects in meteorological research is [6], showing a feasibility study for Cloud computing with a coupled atmosphere-ocean model.

In this paper, we discuss advantages and disadvantages of Cloud and Grid computing for meteorological research, show some computer science issues, and present three examples of meteorological applications, which we have developed for different kinds of distributed computing over the past few years: projects *MeteoAG* and *MeteoAG2* for a Compute Grid and latest *RainCloud* for Cloud computing. We look at issues and benefits mainly from the perspective of system users of distributed computing. Additional or different aspects may apply for service providers and the middleware developer.

II. Aspects of Distributed Computing in Meteorology

A. Grid and Cloud computing

Our practical experience in Grid computing come from projects *MeteoAG* and *MeteoAG2* within the national effort AustrianGrid (AGrid) utilizing the ASKALON environment [7] (see section II-D), including partners and supercomputer centres distributed over Austria [8]. AGrid Phase 1 started in 2005 and concentrated on research of basic Grid technology and application. Phase 2, started in 2008, continued to build on research of Phase 1 and additionally tried to make AGrid self-sustaining. The research aim of this project was not to

 DOI: https://doi.org/10.1109/CloudNet.2014.6968965

develop conventional parallel applications that can be executed on individual Grid machines but to unleash the power of the Grid for single distributed program runs. To simplify this task, all Grid sites are required to run a similar Linux operating system. At the height of the project it consisted of 9 clusters distributed over 5 locations in Austria.

For Cloud computing, plenty of providers offer services, e.g. Microsoft Azure or Google Compute Engine. Those overs can be seen as Platform as a Service (PaaS) solutions, which have the big disadvantage that applications need to be ported to the specific platform to allow execution. The Cloud computing project *RainCloud* uses Amazon Web Services (AWS), simply because it is the most well known and widely used Cloud provider. Their Infrastructure as a Service (IaaS) offer is called Amazons Elastic Compute Cloud (EC2) and offers virtual machines with root access to the users, allowing easy installation of software, libraries and are extreme flexible to use for all kinds of legacy applications. AWS offers additional services for computing, data storage (Simple Storage Service S3) and data transfer (Content distribution networks called Cloudfront), as well as tools for monitoring and planning (Cloudwatch). The services most interesting for scientific computing is EC2 as it allows custom Linux images and kernels allowing to execute all kinds of application: C, C++, Fortran, Java, Shell-Scripts and python.

So called *instances* (i.e.virtual computers) are defined according to their compute power relative to a reference CPU, available memory, storage and network performance. One Elastic Compute Unit (ECU) provides the equivalent CPU capacity of a 1.0-1.2 GHz 2007 Opteron or 2007 Xeon processor. Network performance for the different instance types is specified with the terms: Very low, Low, Moderate, High and 10 Gibabit. Memory and storage are defined in gigabytes and some instances offer solid state disks (SSD) for higher performance.

Figure 1 shows the basic structure of IaaS based Cloud computing on the right side and AGrid as Grid example on the left side. A user might provision multiple (M) instances from a Cloud provider, each with a specific number of CPUs (s-r) and cores running a Linux image of his choice. For Grid resources the offered operating system is out of the end users control. An additional layer, so called Middleware, is applied between the compute resource itself and the end user. The Middleware handles all necessary scheduling, file transfer and setup of Cloud nodes. The end users interact with the Middleware via a Graphical User Interface (GUI) for workflow creation and execution. Execute Engine, Scheduler and Resource Manager interact to effectively use the available resources from available Grids and Clouds and react to changes in the provided computing infrastructure or the executed dynamic workflows.

In the following sections, we list advantages and disadvantages of Grid and Cloud computing, which affected our research most. A general comparison with all vital issues can be found in [9]. Security issues did not apply to this research and operational setting as no sensible data is used. However, for big and advanced operational weather forecasting this might be an issue due to the monetary value of utilized weather sensor data. If security is a concern detailed discussions can be found in [10] for Grid computing, [11] and [12] for Cloud computing.

1) Advantages/Disadvantages Grid:

+ **Handle massive amounts of data**. The full atmospheric model in MeteoAG generated large amounts of data. Through Grid tools like *gridftp* [13] we were able to efficiently transfer and store all simulation data.

+ **Access to HPC which suits parallel applications (e.g. MPI).** The model used in MeteoAG, as many other meteorological models, is a massive parallel application parallelized with MPI. On Grids they run efficiently, however not across different HPC clusters as latencies between distributed data centres can be too high. A middleware can leverage the advantage of access to multiple machines and run applications on suitable machines and parts of workflows in parallel.

- **Different hardware architectures.** During tests in MeteoAG we discovered problems due to different hardware architectures, which can be substantial [14]. We tested different systems with exactly the same setup and software and got consistently different results. In this case this affected our complex full model, but not our simple model. The exact cause is unclear, but most likely a combination of programming, the used libraries (and their versions) and setup (kernel versions) down to the hardware level.

- **Difficult to setup and maintain as well inflexible handling.** The process of getting necessary updates, patches or special libraries needed in meteorology was complex and lengthy or sometimes even impossible due to operating system limitations of the different administrative domains.

- **Special compilation of source code.** For best performance the executables in MeteoAG needed to be compiled for each architecture, with possible side effects. Even in a tightly managed project as AGrid we had to supply three different executables for the meteorological model compiled for the different compatible CPU architectures to avoid significant performance looses.

The Grid offers a *limited amount of resources* but for the executed workflows they were always enough free resources to simulate our models. As part of the Austrian Grid the access to the available resources was available and sufficient as this Grid is used for research mostly and hardly any production runs are executed there.

2) Advantages/Disadvantages Cloud computing:

+ **Cost**. Costs can easily be determined and planned and more details about this important factor are given in section IV.

+ **Full control of software environment, including operating system (OS) with root access**. This proofed to be one of the biggest advantages for our workflows. It is easy to choose the right kernel and version, install needed software, special libraries or modify any component of the system. Cloud providers usually offer standard UNIX operating systems as *images/AMI*, but tuned images can also be saved permanently and made

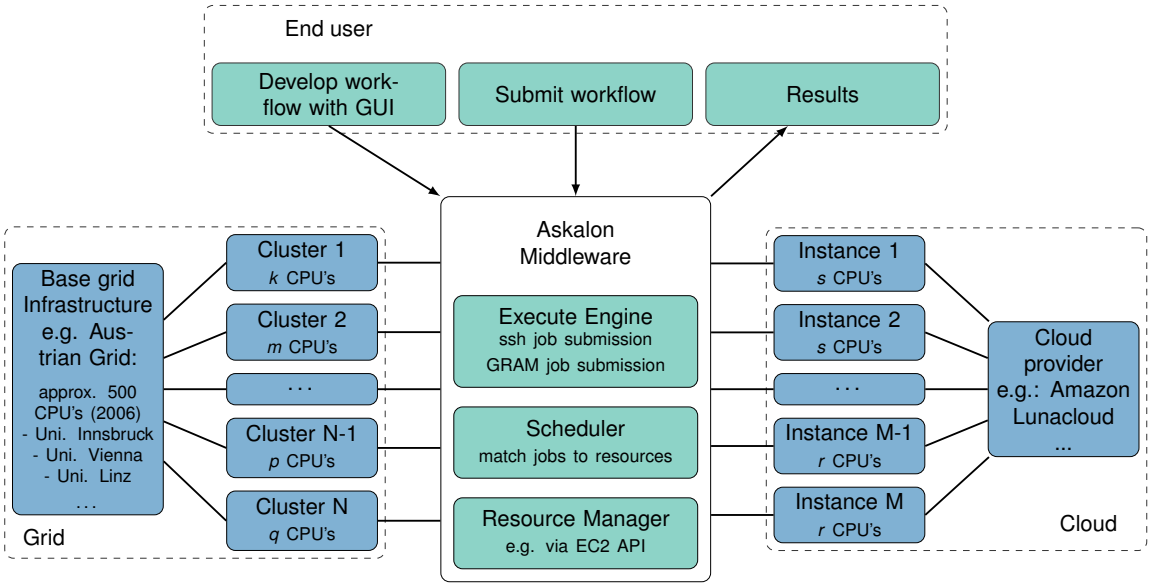

Fig. 1. Schematic setup of the computing environment for Cloud and Grid computing.

publicly available (with additional monthly storing costs).

+ **Simple on demand self service.** For applications with varying requirements for compute power or with daily but short needs for compute power, this is an important characteristics. The resources for the workflow runs were always available within short times, usually the standard on-demand Linux instances were up and running within 5-10 seconds (Amazons documentation states a maximum of 10 minutes).

- **Slow data transfer** Data transfer to and from the Cloud are slow as internet connections are involved. The AGrid was build on University sides sharing the same ACONet backbone, allowing low latency and high bandwidth even between different locations. Additional a higher network latency between most instance types is given, resulting in additional overheads when using this (cheaper) instance types.

Missing information of underlying hardware has no impact on our workflow, as we are not trying to optimize a single model execution. *No common standard between clouds* and *a Cloud provider going out of business* is unimportant for the used application as the used software relies on common protocols like ssh and adaptation to a new cloud provider could be done easily changes to the Middleware configuration.

B. Computer science challenges

For meteorology, as with wide spectrum of scientific applications, the time required to solve a problem grows quickly with problem size (NP complete, [15]). Scheduling different parts of a distributed/workflow application onto heterogeneous resources is one of those problems, which becomes even more complex when multiple parameters are targeted for optimization. Execution time, power consumption, cost, resource utilization and fairness are targets that are considered in optimization processes.

Cloud resource usage adds an additional complexity to the problem as the resource pool no longer is static as in Grids. Cloud providers offer wide range of resource types with different performance characteristics and prices with certain billing intervals. Provisioning the optimal set of resources for the scheduling mechanism is difficult. Executing an application within half of the billing interval (i.e. 30 minutes) leads to the same costs as when running for the full hour. We optimize the scientific application runtime to fit into the billing interval as close as possible. Exceeding that time window by just a few seconds would raise the execution cost significantly and needs to be avoided in all cases (i.e. by a factor of two if the planed execution time was 60 minutes).

C. Solutions

To account for the heterogeneity and loosely coupled nature of resources from Grid and Cloud providers, computer science adopted a workflow paradigm based on loosely coupled coordination of activities. Distributed applications are split in reasonably small execution parts, which can be executed in parallel on distributed systems, allowing the runtime system to optimize resources usage, file transfers, load balancing, reliability, scalability and handle failed parts.

Utilizing the Cloud resource model with coarse grain (hourly) billing intervals results in a new target for optimization: *close to full hour computation*. With a given budget, it is possible to calculate the available resources per planned execution. An optimization target is to scale the model resolution or domain size to fit within the given limit as well as possible. This results in the most efficient results for the given monetary constraints but requires predictable runtime of the overall workflow execution.

D. Middleware ASKALON

To make it as simple as possible for a (none computer) scientist to use distributed computing resources, we make use of a so called Middleware system. ASKALON, an existing

Middleware from the Distributed and Parallel Systems group in Innsbruck, provides integrated environments to support the development and execution of scientific workflows on dynamic Grid and Cloud environments [7].

Figure 1 shows the design of the ASKALON system. Workflows can be generated in a scientist-friendly GUI and submitted for execution to a service to allows long lasting workflows without the need for the user to be online throughout the whole execution period.

Three main components handle the execution of the workflow:

Scheduler Activities are mapped to physical (or virtualized) resources for their execution. A wide set of scheduling algorithms is available e.g. JustInTime or DCP-C [16]. For a reasonable scheduling result, a prediction service providing estimated activity runtimes is utilized [17].

Resource Manager Cloud resources are known to *scale by credit card* and theoretically an infinite amount of resources is available[1]. The resource manager has the task to provision the right amount of resources at the right moment to allow the execute engine to run the workflow as the scheduler decided. Cost constraints must be strictly adhered to as budgets are in practice limited.

Execute Engine Submission of jobs and transfer of data to the compute resources is done with a suitable protocol, e.g. ssh for Cloud resources or GRAM in a Globus/Grid environment.

E. Open challenges

Improvements to our current system for executing the meteorological workflows targets:

Runtime prediction A precise prediction for execution and file transfer is needed to optimize the runtime to utilize the resources as efficiently as possible. The current solution employs several minutes buffer between execution time and billing interval to avoid increased costs as predictions are not precise enough for Cloud resources and networks.

Dynamic problem size Amazon, for example, is offering resources with variable pricing, which provides the chance of cheaper computation. Assuming a fixed budget for the executions the problem size could be increased dynamically to meet the computational power currently available for the given budget. In such a case a price increase during execution would require dynamic workflow and input parameter changes during the application execution. This would require a more dynamic application implementation and changes of the middleware.

Federated Cloud The current approach utilized Amazon EC2 or a private Cloud as the two possible Cloud provider. Additional Clouds are supported by ASKALON but their simultaneous usage introduces new challenges for the scheduler. Inter-Cloud file transfers may have a unpredictable (slow) performance compared to transfers within a single Cloud. Pricing schemes of different providers are often hard to compared. Ways to see multiple commercial Cloud providers as a one bigger federated Cloud are needed.

[1]Amazon EC2 allows up to 20 instances for a user. If more instances are needed a manual request form has to be filled out to get permanent access to bigger resource pools.

III. Applications in Meteorology

In the following subsections, we describe the three applications we developed for meterological research and used Grid and Cloud resources for faster execution. All projects investigate orographic precipitation over complex terrain. The most important characteristics regarding distributed computing of the projects are shown in table I.

A. MeteoAG

MeteoAG started as part of the AGrid computing initiative. Using ASKALON we created a workflow to run a full numerical atmospheric model and visualization on a Grid infrastructure [18]. The model is the non hydrostatic Regional Atmospheric Modeling System (RAMS), a fully MPI parallelized Fortran based code [19]. NCAR Graphics library is used for visualisation. Due to all AGrid sites running a similar Linux OS, no special code adaptation to Grid computing was needed.

We simulated real cases as well as idealised test cases in the AGrid environment. Most often these are parameter studies testing sensitivities to certain input parameters with many slightly different runs. The investigated area in the realistic simulations covers Europe and a target area over western Austria, with resolution of the innermost domain of 500m and 60 vertical levels (approx. 7.5 million grid points). Figure 2 shows the workflow deployed to the AGrid. Starting with many simulations with a shorter simulation time, it was then decided which runs to extend further. Only runs where heavy precipitation occurs were chosen. Post-processing done on the Compute Grid includes extraction of variables and preliminary visualization, but the main visualization is done on a local machine.

The workflow characteristics relevant for distributed computing are: fewer model instances but highly CPU demanding as well as lots of interprocess communications. Results of this workflow require a substantial amount of data transfer between the different Grid sites and the end-user.

Upon investigation of our first runs it was necessary to provide different executables for specific architectures (32bit, 64bit, 64bit Intel) to get optimum speed.

B. MeteoAG2

MeteoAG2 is the continuation of MeteoAG and also part of AGrid [20]. Based on the experience of MeteoAG that it is much more effective to deploy an application consisting of serial CPU jobs, MeteoAG2 uses a simpler meteorological model, the Linear Model of orographic precipitation (LM) [21]. The model computes only very simple linear equations of orographic precipitation, is not parallelized, and has short runtime, in the 10 seconds area, even with high resolutions (500m) over large domains. LM is written in Fortran. ASKALON is used for workflow execution and Matlab routines for visualisation of the output files that the workflow produces.

With this workflow, rainfall over the Alps was investigated by taking input from European Centre for Medium-Range

TABLE I. OVERVIEW OF OUR PROJECTS AND THEIR WORKFLOW CHARACTERISTICS.

Project	MeteoAG	MeteoAG2	RainCloud
Type	Grid	Grid	Cloud
Meteorological model	RAMS (Regional Atmospheric Modeling System)	single layer linear model of orographic precipitation	double layer linear model of orographic precipitation
Model type	complex full numerical model parallelized with Message Passing Interface (MPI)	simplified model	double layer simplified model
Parallel runs	20-50	approx 50000	$>$ 5000 operational, $>$ 10000 research
Runtime	several days	several hours	1-2 hours operational / $<$ 1h research
Data transfer	200GB	1GB	10MB - 1GB
Workflow flexibility	strict	strict	flexible
Applications	parameter studies, case studies	downscaling	parameter studies, downscaling, probabilistic forecasts, model testing
Intent	research	research	operational, research
Frequency	on demand	on demand	operational: daily, research: on demand
Programming	shell scripts, Fortran, NCAR Graphics, MPI	shell scripts, Fortran, Matlab	python, Fortran

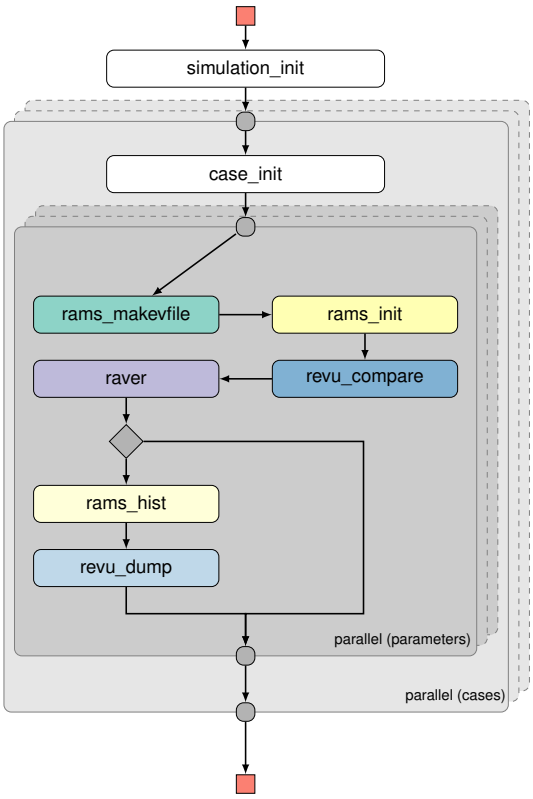

Fig. 2. Workflow of MeteoAG using the Regional Atmospheric Modelling System (RAMS) and supporting software REVU (extracts variables) and RAVER (analyses variables).

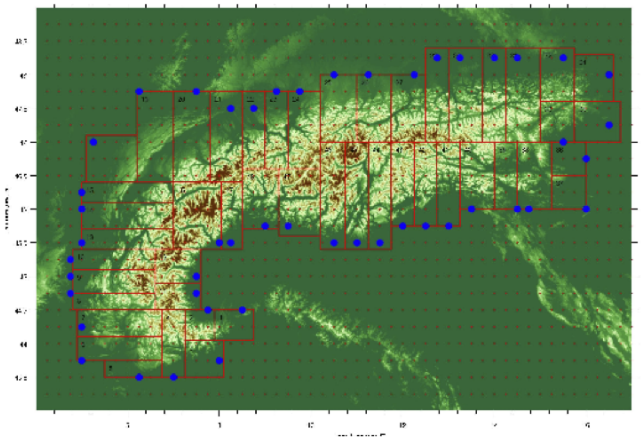

Fig. 3. Grid setup of experiments in MeteoAG2.

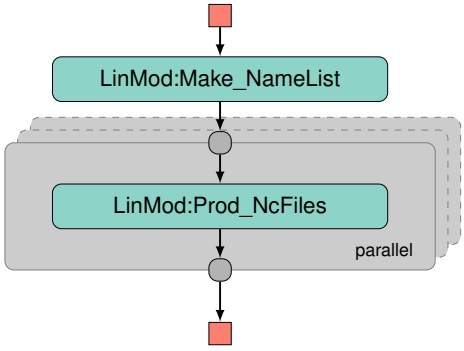

Fig. 4. Workflow of MeteoAG2 using the Linear Model (LM) of orographic precipitation.

Weather Forecasts (ECMWF) model, splitting the Alps into subdomains (see figure 3) and running the model within each subdomain with variations in the input parameters. The last step combines the results from all subdomains and visualises them. Using Grid computing allowed us to run many over 50.000 simulations in a relatively short amount of time (several hours).

The workflow deployed to the Grid (figure 4) is simple with only two main activities: preparing all the input parameters for all subdomains and then the parallel execution of all runs. One of the drawbacks of MeteoAG2 is the very strict setup that was necessary due to the state of ASKALON at that time, e.g. no robust if-construct yet, and the direct use of model executables without wrappers. The workflow could not easily be changed to suit different research needs, e.g. change to different input parameters for LM or using a different model. Ongoing developement and bugfixes in the middleware later on allowed to creat more flexible workflows as shown in the next use case.

C. RainCloud

Switching to Cloud computing, RainCloud uses an extended version of the same simple model of orographic precipitation as MeteoAG2. The main extension to LM is the ability to simulate different layers, while still retaining its fast

execution time [22]. The software stack includes ASKALON again, the Fortran-based LM, python scripts and Matpotlib for visualisation.

The inclusion of if-constructs in ASKALON and a different approach to the scripting of activities (e.g. wrapping the model executables in python scripts and calling these) allows RainCloud be be used in different setups. It was possible to create one generic workflow that can be used for multiple use cases by making parts of the workflow optional using if-structures and boolean input parameters to enable or disable those parts. We are now able to run the workflow in 3 flavours without any changes: idealised, semi-idealised and realistic simulations as well as different settings: operational and research. Figure 5 depicts the workflow run on Cloud computing. Only the first two activities, *PrepareLM* and *LinearModel* have to be run, the others are optional. This workflow fits a lot of meteorological applications as it has the building blocks:

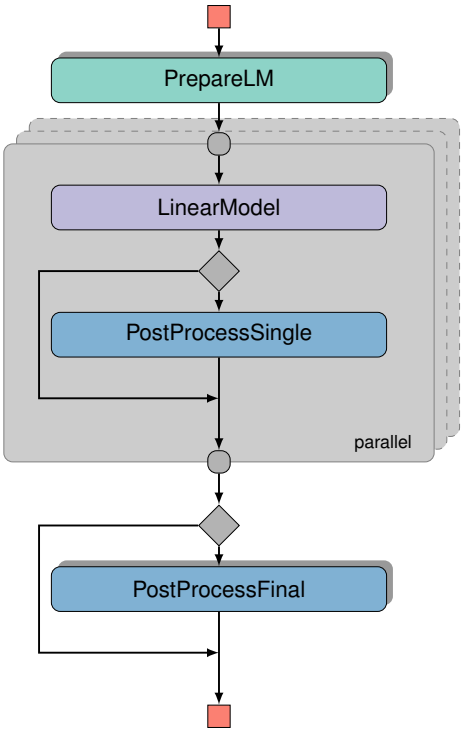

Fig. 5. Workflow of RainClouds operational setting for the Avalanche Warning Service Tyrol (LWD)

- preparation of the simulations (*PrepareLM*)
- execution of a meteorological model (*LinearModel*)
- post processing of each individual run, e.g. for producing derived variables (*PostProcessSingle*).
- post processing of all runs (*PostprocessFinal*).

The operational setup produces spatially detailed daily probabilistic precipitation forecasts for the Avalanche Service Tyrol (Lawinenwarndienst Tirol) to help forecast avalanche danger. Figure 5 shows the workflow.

Our workflow invocations vary substantially in required computation power as well as data size. The operational job is run daily during winter, whereas research types are run in bursts. Data usage *within* the Cloud can be substantial in the area of 500Gb with all flavours, but with big differences of data transfer from the Cloud back to the local machine. Operational results are small, in the order of 100Mb, while research results can amount up to 100Gb, influencing the overall runtime due to the additional transfer time as well as the costs.

IV. Costs, performance and usage scenarios

To define the exact costs for a dedicated server system or the participation in a Grid initiative is not trivial, and often even unknown to the provider. We contacted several of them, but due to complicated budgeting the final costs are not obvious. [23] discusses costs for operating a server environment for data services from a providers perspective. Costs shown are servers, infrastructure, power requirement and networking, however not mentioning cost of human resources for e.g. system administration. [24] include human resources and establish a cost model for setup and maintenance of a data center. Grids may have different and negotiable levels of access and participation, with varying associated costs.

Cloud computing on the other hand offers simpler and transparent costs. Pricing varies depending on the provider, capability of a resource, but also on the geographical region. For example Amazon currently offers centres in the US, EU, Asia Pacific and South America resulting in different transfer speeds towards the users location. Prices of AWS *on-demand* compute instances for Linux OS can be found in table II and are 0.02 USD/hour up to ~5 USD/hour (region Ireland).

Cheaper instance pricing is available through *spot* instances with which one bids on spare resources. These resources might get cancelled if demand rises, but are a valid option for interruption-tolerant workflows or for developing a workflow.

Figure 6 shows the difference between spot and on-demand pricing for 25 test runs of our operational workflow (circle and x, right y-axis). All runs use a total of 32 cores but a different number of instances. Runtime only includes the actual workflow, not the spin up needed to prepare the instances. It usually takes 5-10 seconds for an instance to become available. Spot and on-demand only differ in the pricing scheme not in the computational resources themselves. With spot pricing we achieved savings between 65-89 percent, however with an additional startup latency of 2-3 minutes (compared to 5-10 seconds) as spot instances need longer for instantiation.

Additional costs for data transfer from and to the Cloud apply but the RainCloud workflow incurred no additional data transfer costs as AWS has an allotment of 1GB/month free transfer which was never exhausted.

A simple cost comparison can be done with the purchasing costs of dedicated hardware, excluding costs for system administration, cooling or power. The operational part of RainCloud runs on 32 cores for approximately 3h per day during winter half year, i.e. 550h per year. A standard 4 core desktop PC with 8 GB RAM costs around 850 USD, equaling to 3500h of a c3.xlarge AWS instance (4 core, 7.5 GB RAM, 0.239USD/h). A dedicated 32 core server with 64GB RAM costs around 5500 USD (various brands, excluding Austrian taxes). A comparable on-demand AWS instance (c3.x8large; 32 cores, 60 GB RAM)

TABLE II. PRICES AND SPECIFICATIONS FOR AMAZON EC2 ON DEMAND INSTANCES RUNNING LINUX OS IN REGION *EU-west* AS OF JUNE 2014.

Instance Family	Instance Type	vCPU	ECU	Memory (GiB)	Storage (GB)	Cost USD/h
General purpose	m3.medium	1	3	3.75	1 x 4 SSD	0.077
General purpose	m3.xlarge	4	13	15	2 x 40 SSD	0.154
General purpose	m3.2xlarge	8	26	30	2 x 80 SSD	0.616
Compute optimized	c3.large	2	5	3.75	2 x 16 SSD	0.120
Compute optimized	c3.xlarge	4	14	7.5	2 x 40 SSD	0.239
Compute optimized	c3.8xlarge	32	108	60	2 x 320 SSD	1.912
Memory optimized	r3.large	2	6.5	15	1 x 32 SSD	0.195
Memory optimized	r3.2xlarge	8	26	61	1 x 160 SSD	0.780
Memory optimized	r3.8xlarge	32	104	244	2 x 320 SSD	3.120
Storage optimized	i1.4xlarge	16	53	122	4 x 800 SSD	3.751
Storage optimized	hs1.8xlarge	16	35	117	24 x 2048	4.900
Micro instances	t1.micro	1	Var	0.615	EBS only	0.020
GPU instances	c2.2xlarge	8	26	15	1 x 60 SSD	0.702
General purpose	m1.xlarge	4	8	15	4 x 420	0.520
General purpose	m1.medium	1	2	3.75	1 x 410	0.130
Memory optimized	m2.4xlarge	8	26	68.4	2 x 840	1.840

could run for ~2800 hours at 1.91USD/h pricing. Assuming no instance price variance, our operational workflow could be run for approximately five years, the usual depreciation time for hardware.

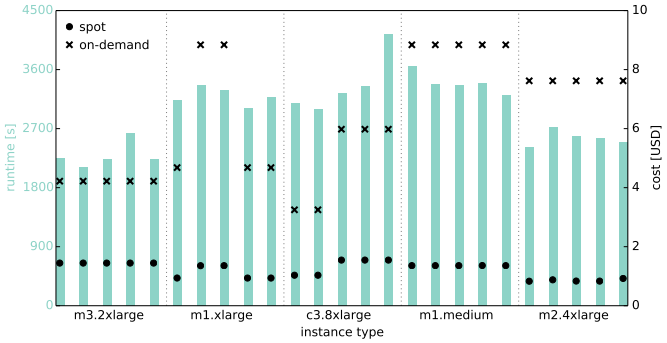

Fig. 6. Overall runtime of one operational run on various EC2 instance types.

Figure 6 shows the effect of different instance types on the runtime of our operational RainCloud workflow. Dots show costs for on-demand instances (x) and spot instances (circle; right y-axis). First, a clear difference between the instance types shows, longest running taking nearly twice as long as the shortest one. Second, even within one instance type, runtime varies between 10-20 percent. Serial execution on a one core desktop PC takes about 12h, i.e. a speedup of ~18. Based on these experiments our daily operational workflow uses four m3.2xlarge instances.

[14] show speedup and performance for MeteoAG with various problem sizes. Speedup of multiple cores versus 1 core for a short running test setup is ~5, with higher speedups possible for a full complex workflow run. For MeteoAG2 [20] show a speedup of ~120 of executing the workflow on several Grid machines compared to the execution on a single desktop PC. However, as these are different workflows, no comparison between the type of computing resources can be made from these performance measures.

In meteorology different usage scenarios are commonly found. For choosing the right type of computing system several issues need to be taken into account. Only above a certain workflow length moving away from a local machine is worth the effort. Grids usually have a steep learning curve while Clouds offer simple (web) interfaces To make the most out of Cloud computing (and to some extent out of Grid computing) it is best to have a workflow which can be split into small, independent components and utilize a Middleware like ASKALON for the execution, that reduces the complexity for scientists.

For a research scenario with bursts of high activity with many but small activities, Cloud computing fits perfectly. The costs are fully controllable and only little setup is required. Examples are parameter studies with simple models, computation of model output statistics (MOS) or satellite data processing. If a lot of data transfer is needed Grid computing is the better alternative. Research applications with big, long running, data intensive simulations such as high resolution complex models are best run on Grids or local clusters.

In an operational scenario with frequent invocations, Clouds and Grids might be suitable depending on the amount of data transferred. For simple models or preprocessing of data Clouds offer a cheap alternative. However, for full forecast models dedicated local cluster are usually the fastest and most reliable option. Time critical data dissemination of forecasts can be sped up with Grids. Operational scenarios with infrequent invocations might benefit from using Grid or even Cloud computing, avoiding the need for a local cluster. Examples are recalculation/reanalysis of seasonal/climate simulations or updating of MOS equations.

V. CONCLUSION

We successfully executed meteorological applications on distributed computing infrastructure. Grids and Clouds resources bouth showed advantages and disadvantage which we analysed. Our meteorological applications range from a complex atmospheric limited-area model to a simplified model of orographic precipitation. Adhering to some limitations/considerations, distributed computing can cater to both.

If Grid is seen as an agglomeration of individual clusters, complex parallelized models are simple to deploy and efficient to use in a research setting. The compute power is usually substantially larger than what a single institution could afford. However, in an operational setting the immediate availability of resources might not be given. This is an issue that needs

to be addressed in advance. For data storage and transfer, e.g. dissemination of forecasts, Grids are a powerful tool.

Taking Grid as a structure, workflows involving MPI are not simple to exploit. As with Clouds, it is more efficient to deploy an application consisting of serial jobs with as little interprocess communication as possible.

The setup and access to Cloud infrastructure is a lot simpler and less effort than participation in a Grid project. Grids require hardware and more complex software to access whereas access to Clouds is usually kept as simple as possible. A Credit card and one time reachability via phone to verify a AWS account is the only requirement to get access to the theoretical unlimited resources offered by Amazon. (Commercial) Cloud computing is very effective and cost saving tool for certain meteorological applications. Individual projects with high-burst needs or an operational setting with a simple model are two examples.

The biggest disadvantages of Clouds are data transfer to and from the Cloud. Within the Cloud infrastructure the transfer speeds are comparable to local resources but over internet connections, especial from Europe to America, transfers are considerably slower than for a dedicated cluster setup or Grids.

Private Clouds remove some of the disadvantages of public Clouds, security and data transfer are the most notable ones. However, using private Clouds also removes the advantage of not needing hardware and system administration. We used a small private Cloud to develop our workflow before going full-scale on Amazon AWS with our operational setup.

In a meteorological research setting with specialised software, Clouds offer a flexible system with full control over operating system, installed software and libraries. Grids on the other hand are managed on individual Grid sites and are more strict and less flexible. The same is valid for customer service. Clouds offer one contact for all problems and offer (paid) premium support as opposed to having to contact each system administration for every Grid site.

We showed that meteorological application can be executed using Grid and Cloud resources and depending on the application and its usage patterns either Grid or Cloud fits the end user needs more.

REFERENCES

[1] I. Foster and C. Kesselman, *The Grid 2: Blueprint for a new computing infrastructure*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2003.

[2] I. T. Foster and C. Kesselman, *The Grid: Blueprint for a new computing infrastructure*, 2nd ed. Amsterdam: Morgan Kaufmann, 2004.

[3] M. L. Bote-Lorenzo, Y. A. Dimitriadis, and E. G. A. Sanchez, *Grid Characteristics and Uses: A Grid Definition*. Springer Berlin / Heidelberg, 2004, vol. 2970.

[4] V. Nefedova, R. Jacob, I. Foster, Z. Liu, Y. Liu, E. Deelman, G. Mehta, M.-H. Su, and K. Vahi, "Automating Climate Science: Large Ensemble Simulations on the TeraGrid with the GriPhyN Virtual Data System," *e-science*, vol. 0, p. 32, 2006.

[5] P. Mell and T. Grance, "The NIST definition of cloud computingrecommendations of the National Institute of Standards and Technology. Special Publication 800-145, NIST, Gaithersburg," *csrc. nist. gov/publications/nistpubs/800-145/SP800-145. pdf*, vol. 15, no. 10.07, p. 2009, 2011.

[6] C. Evangelinos and C. Hill, "Cloud Computing for parallel Scientific HPC Applications: Feasibility of running Coupled Atmosphere-Ocean Climate Models on Amazon's EC2." *ratio*, vol. 2, no. 2.40, pp. 2–34, 2008.

[7] S. Ostermann, K. Plankensteiner, R. Prodan, T. Fahringer, and A. Iosup, "Workflow monitoring and analysis tool for ASKALON," in *Grid and Services Evolution*, Barcelona, Spain, June 2008, pp. 73–86.

[8] J. Volkert, "The Austrian Grid Initiative - High Level Extensions to Grid Middleware," in *PVM/MPI*, ser. Lecture Notes in Computer Science, D. Kranzlmüller, P. Kacsuk, and J. J. Dongarra, Eds., vol. 3241. Springer, 2004, p. 5.

[9] I. Foster, Y. Zhao, I. Raicu, and S. Lu, "Cloud Computing and Grid Computing 360-Degree Compared," *2008 Grid Computing Environments Workshop*, pp. 1–10, Nov. 2008.

[10] E. Cody, R. Sharman, R. H. Rao, and S. Upadhyaya, "Security in grid computing: A review and synthesis," *Decision Support Systems*, vol. 44, no. 4, pp. 749–764, Mar. 2008.

[11] D. Catteddu, "Cloud Computing: Benefits, Risks and Recommendations for Information Security," in *Web Application Security SE - 9*, ser. Communications in Computer and Information Science, C. Serrão, V. Aguilera Díaz, and F. Cerullo, Eds. Springer Berlin Heidelberg, 2010, vol. 72, p. 17.

[12] D.-G. Feng, M. Zhang, Y. Zhang, and Z. Xu, "Study on cloud computing security," *Journal of Software*, vol. 22, no. 1, pp. 71–83, 2011.

[13] B. Allcock, J. Bester, J. Bresnahan, A. L. Chervenak, I. T. Foster, C. Kesselman, S. Meder, V. Nefedova, D. Quesnel, and S. Tuecke, "Data management and transfer in high-performance computational Grid environments," *Parallel Computing*, vol. 28, no. 5, pp. 749–771, 2002.

[14] F. Schüller, J. Qin, F. Nadeem, R. Prodan, T. Fahringer, and G. Mayr, "Performance, Scalability and Quality of the Meteorological Grid Workflow MeteoAG," *Austrian Computer Society*, vol. 221, pp. 155–165, 2007.

[15] M. R. Garey and D. S. Johnson, *Computers and Intractability / A Guide to the Theory of NP-Completeness*. San Francisco: W.H. Freeman and Company, 1978.

[16] S. Ostermann and R. Prodan, "Impact of variable priced cloud resources on scientific workflow scheduling," in *Euro-Par 2012 Parallel Processing*, ser. Lecture Notes in Computer Science, C. Kaklamanis, T. Papatheodorou, and P. Spirakis, Eds., vol. 7484. Springer Berlin Heidelberg, 2012, pp. 350–362.

[17] F. Nadeem and T. Fahringer, "Predicting the execution time of grid workflow applications through local learning," *High Performance Computing Networking, Storage and Analysis, Proceedings of the Conference on*, vol. 1, no. 1, pp. 1–12, 2009.

[18] F. Schüller, *Grid computing in meteorology: grid computing with - and standard test cases for - a meteorological limited area model*. Saarbrücken, Germany: VDM, 2008.

[19] W. R. Cotton, R. A. P. Sr., R. L. Walko, G. E. Liston, C. J. Tremback, H. Jiang, R. L. McAnelly, J. Y. Harrington, M. E. Nicholls, G. G. Carrio, and J. P. McFadden, "RAMS 2001: Current status and future directions," *Meteorology and Atmospheric Physics*, vol. 82, no. 1, pp. 5–29, Jan. 2003.

[20] K. Plankensteiner, J. Vergeiner, R. Prodan, G. Mayr, and T. Fahringer, "Porting LinMod to Predict Precipitation in the Alps using ASKALON on the Austrian Grid," in *3ˆrd Austrian Grid Symposium*, J. Volkert, T. Fahringer, D. Kranzlmüller, R. Kobler, and W. Schreiner, Eds., vol. 269. Austrian Computer Society, 2009, pp. 103–114.

[21] R. B. Smith and I. Barstad, "A linear theory of orographic precipitation," *Journal Of The Atmospheric Sciences*, vol. 61, no. 12, pp. 1377–1391, Jun. 2004.

[22] I. Barstad and F. Schüller, "An Extension of Smiths Linear Theory of Orographic Precipitation: Introduction of Vertical Layers," *Journal of the Atmospheric Sciences*, vol. 68, no. 11, pp. 2695–2709, Nov. 2011.

[23] A. Greenberg and J. Hamilton, "The cost of a cloud: research problems in data center networks," *ACM SIGCOMM Computer Communication Review*, vol. 39, no. 1, pp. 68–73, 2008.

[24] C. D. Patel and A. J. Shah, "Cost model for planning, development and operation of a data center," 2005.