



## UvA-DARE (Digital Academic Repository)

### A user-centric execution environment for CineGrid workloads

Dumitru, C.; Grosso, P.; de Laat, C.

**DOI**

[10.1016/j.future.2015.03.021](https://doi.org/10.1016/j.future.2015.03.021)

**Publication date**

2015

**Document Version**

Final published version

**Published in**

Future Generation Computer Systems

**License**

Article 25fa Dutch Copyright Act (<https://www.openaccess.nl/en/policies/open-access-in-dutch-copyright-law-taverne-amendment>)

[Link to publication](#)

**Citation for published version (APA):**

Dumitru, C., Grosso, P., & de Laat, C. (2015). A user-centric execution environment for *CineGrid* workloads. *Future Generation Computer Systems*, 53, 55-62.

<https://doi.org/10.1016/j.future.2015.03.021>

**General rights**

It is not permitted to download or to forward/distribute the text or part of it without the consent of the author(s) and/or copyright holder(s), other than for strictly personal, individual use, unless the work is under an open content license (like Creative Commons).

**Disclaimer/Complaints regulations**

If you believe that digital publication of certain material infringes any of your rights or (privacy) interests, please let the Library know, stating your reasons. In case of a legitimate complaint, the Library will make the material inaccessible and/or remove it from the website. Please Ask the Library: <https://uba.uva.nl/en/contact>, or a letter to: Library of the University of Amsterdam, Secretariat, P.O. Box 19185, 1000 GD Amsterdam, The Netherlands. You will be contacted as soon as possible.



## A user-centric execution environment for *CineGrid* workloads

Cosmin Dumitru\*, Paola Grosso, Cees de Laat

System and Network Engineering research group, Informatics Institute, University of Amsterdam, Netherlands



### HIGHLIGHTS

- We describe the general challenges user face in heterogeneous it infrastructure environments and identify their instance in the case of *CineGrid*.
- We characterize the most frequent workloads in the *CineGrid* collaboration including their infrastructure resource requirements.
- We present the design and implementation of a system which simplifies workload resource definition by leveraging semantic web description languages for infrastructure.
- We provide Pareto-optimal resource configurations for workload execution.

### ARTICLE INFO

#### Article history:

Received 1 July 2014

Received in revised form

16 December 2014

Accepted 26 March 2015

Available online 4 June 2015

#### Keywords:

Cinegrid

Resource selection

Virtual infrastructure

Semantic web

Bags-of-tasks

### ABSTRACT

The abundance and heterogeneity of IT resources available, together with the ability to dynamically scale applications poses significant usability issues to users. Without understanding the performance profile of available resources users are unable to efficiently scale their applications in order to meet performance objectives. High quality media collaborations, like *CineGrid*, are one example of such diverse environments where users can leverage dynamic infrastructures to move and process large amounts of data. This paper describes our user-centric approach to executing high quality media processing workloads over dynamic infrastructures. Our main contribution is the *CGtoolkit* environment, an integrated system which aids users cope with the infrastructure complexity and large data sets specific to the digital cinema domain.

© 2015 Elsevier B.V. All rights reserved.

### 1. Introduction

Federated resources enable collaboration and allow for more flexibility when it comes to demanding workloads. Federated resources can be data, like in the case of the *CineGrid Exchange* [1], a repository dedicated to high quality media, data and compute resources as in the case of the Open Science Data Cloud [2] or generic compute and network resources like in the case of the GENI/ExoGENI [3] project. In addition to the diverse federated resources users can also use on-demand resources, like the one provided by cloud computing service providers such as Amazon Elastic Cloud Compute (Amazon EC2) [4].

Given so much flexibility and heterogeneity the users are left with a lot of infrastructure complexity to manage. This complexity is increased by the users' applications which have their own specific requirements, for instance a specific data set, Quality of Service(QoS) requirements, or metric related requirements: cost, expected performance, CO2 footprint of the used hardware, and so

on. When users design and run their distributed applications they must have all these parameters in mind and therefore should select the resources accordingly.

It is difficult to select the right set of resources for application execution. Under-provisioning a resource type, such as network capacity in a distributed environment, affects performance in a negative way. Over-provisioning on the other hand, potentially impacts other users which share the global pool of resources or incurs unnecessary costs, and ideally should be avoided. While manual resource selection is a valid approach, it has the potential of requiring multiple iterations until the user constraints are satisfied. However, not all users are experts and most of them just want to use the resources as a utility and do not want to be bothered with understanding the complex processes which hide behind the infrastructure and applications. Therefore we argue that the task of resource selection should be offloaded as much as possible to a capable automated system.

Our work takes place in the context of *CineGrid* [5], a community that exploits the recent advances in computing and network infrastructures and adapts them to the digital cinema world. *CineGrid* mission is "To build an interdisciplinary community that is focused on the research, development, and demonstration of networked collaborative tools to enable the production, use,

\* Corresponding author.

E-mail address: [c.dumitru@uva.nl](mailto:c.dumitru@uva.nl) (C. Dumitru).

preservation and exchange of very high quality digital media over photonic networks". The community operates a distributed testbed spanning over multiple continents (Europe, Asia, North and South America). All the sites are connected using high-speed dynamic photonic networks provided by the Global Lambda Integrated Facility community [6]. Each member operates a number of high capacity storage nodes while some also operate visualization and computing facilities that are able to access, display and process the available content(assets). Together the storage nodes form the *CineGrid Exchange* [1], a distributed federated environment that provides increased data security for the high quality multimedia content.

Our main contributions are:

- We identify and characterize the CineGrid computing workloads (Section 2).
- We describe a user-centric system, the *CGtoolkit*, which is able to execute *CineGrid* workloads while abstracting away the resource complexity (Section 3).
- We showcase and evaluate the system in different scenarios (Section 4).

## 2. Motivation

The *CineGrid* workloads are centered around very high quality media related to the digital cinema domain. Given the very high resolution and bit depth, the amount of data for the final version of a video production can easily go in the ranges of hundreds of gigabytes or even terabytes. For example, the 10 min short, 24 frames per second, open source movie *Sintel* takes over 160 GB when stored in an uncompressed image format in 4k(4096 × 1744) resolution using 8 bits for each color channel (RGB). Consequently, the 16 bit per channel version has a total size of approximately 650 GB [7]. A recent trend in digital cinema is to increase the frame rate to 48, 60 or even 120 frames per second and in this case, the storage requirement increases proportionally with the frame rate. For the 3D version of the same clip, the storage requirement doubles. Even more, during the production process of a movie, a sequence can have multiple versions, and each has to be kept until the user decides which one to use. This massive data scale challenges traditional methods and demands new approaches to visualization, processing and delivery methods.

We identify three dominant workloads:

- video streaming
- data replication
- data processing—ingestion (resizing), compression, color correcting, transformations, etc.

The first two workloads are mostly network bound and require fast networks and high capacity storage. The third one also involves considerable compute resources if timeliness is required.

Our key insight is to use theory and methods from the high performance computing (HPC) world for the *CineGrid* workloads which involve image processing. Most of the *CineGrid* related tasks involve processing individual files, the frames of a movie clip. This is common way of storing assets in the digital cinema industry, as video containers, which pack multiple images into a single file, are difficult to transfer and edit. For data locality and available reasons, assets are stored and replicated in *CineGrid* Exchanges, network high capacity storage servers.

A large subset of these processing tasks can be seen as *bags-of-tasks(BoT)* [8], a set of independent tasks which can be executed in any order. Tasks like individual image compression, resizing, filtering (e.g. blurring, sharpening, de-bayering, color space transformation), information extraction (e.g. color histogram, object recognition) can all be seen as tasks from a bag-of-tasks. Besides

these there are also other types of tasks which require a certain ordering or more complex workflow, yet we limit the optimization techniques presented in this paper to the simpler case mentioned at the beginning of this paragraph. One important characteristic is that because the task usually involve very large images, the workload becomes *data-intensive*. One of the consequences of the *data-intensive* property is the inability of the system to scale efficiently. Even if new resources are added, the performance of the system does not increase as I/O becomes the bottleneck. This property needs to be taken into consideration prior to execution if the user has any kind of performance or QoS requirements. This brings the need of a tool which is able to help the user in both selecting and fine tuning the supporting infrastructure to achieve his performance goals.

## 3. System design

In our system, the resource types are semantically annotated using the Infrastructure and Network Description Language (INDL) [9]. INDL provides the basis for infrastructure visualization, that is, the ability to dynamically combine infrastructure elements at runtime. The applications defined by the users can also include specific INDL resource requirements. By leveraging the semantic web annotations, the system is able to automatically select and interact with the correct infrastructure components. For example, in the streaming scenario, the user has to only select the video clip to stream and the receiver. Finding a suitable data source compatible with the receiver, establishing connectivity and execution are all left to the system to handle.

As explained in Section 2, the dominant computing workload for *CineGrid* consists of Bags-Of-Tasks. We designed the system to assist users in the execution of Bags-of-Tasks workloads. It allows users to evaluate the performance of various available infrastructure elements and then make an informed selection for the execution phase.

The *CGtoolkit* consists of two modules:

- The Web Portal—the user interface and content and meta-data repository (Catalog).
- The Execution Engine—manages resource provisioning and application execution.

### 3.1. Web Portal

The Web Portal acts as the user interface of the system. Here users can define their applications and can then execute them. Each application has different resource requirements.

It also includes two data repositories, as shown in Fig. 1. The content available in various Exchanges is described using the CineGrid Description Language (CDL) and the information is stored in the Portal. The content meta-data is split into two parts, one which refers to intrinsic properties of the assets, like authorship/copyright information, creation date and another one which refers to properties that relate to the image properties: file format, resolution, compression options, bit depth, etc.

The second data repository refers to the available resources and their properties. The system administrator can add and remove resource and resource types as they become available. Each resource type implements one or more CDL *services* [10]. A CDL *service* can consume data stored in the exchanges. For instance, a high capacity storage node implements the *Streamer* and *Storage* services while a node which has 4k capable display will implement a *Visualizer* service. By decoupling the functionality from the physical representation (the server connected to the network), the system can dynamically couple services at runtime. Fig. 2 presents a subset of the *CineGrid* Amsterdam Exchange described using

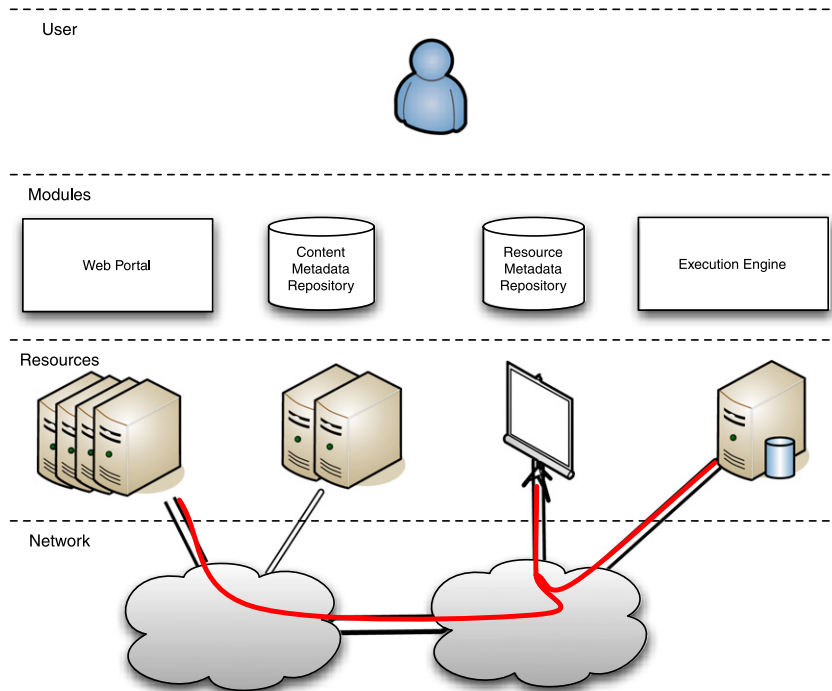


Fig. 1. System architecture.

CDL. It shows one of the CineGrid Amsterdam Exchange storage nodes connected to one of the sites of the DAS4 distributed cluster. These two resources can be used for the storage, streaming and processing. The CGEX node provides two services: storage and visualization (receiver for a video stream). The compute cluster also provides two types of services: storage and processing. The cluster provides also a transcoding service that can encode and decode to and from different types of image formats.

Users can view the currently available content stored on the various storage nodes the *CineGrid* Exchange. Fig. 3 shows the Catalog view of the portal. Content can be located using various criteria like resolution, available format, physical location, etc.

In the case of video streaming, the system first finds suitable sources for the video asset selected by the user. Fig. 4 presents the content detail view of the portal, which includes content meta-data information and a browser-based low resolution clip preview. Next, if needed, the system provisions a network path from the source storage node to the video stream receiver. The receiver can be a projector, dedicated display or just a storage service. Once network connectivity has been established the user can start streaming the video content. In Fig. 5 we present the playlist feature of the portal. It allows users to stream video content in sequence to the same receiver. Network connections are provisioned at runtime for each video asset which will be streamed. All the information required to provision the network paths is extracted from the resource repository and is used then to invoke a capable network provisioning system, like OpenNSA [11], an implementation of the Network Service Architecture, a unified API for multidomain networks.

### 3.2. Execution engine

The Execution Engine uses a master-worker model in which task are selected randomly from the bag-of-tasks and are then submitted to the workers in a self-scheduling manner. That is, when a worker completes a job, it will be assigned a new one from the bag. In literature this scheduling policy is also known as *Demand-Driven* [12]. This job scheduling policy is one of the many

existing policies. We plan to explore the behavior of the system with other policies in future work. As the order is random, we can view the execution process as a stochastic process and therefore use specific methods to analyze it. Currently the execution engine assumes that there is only one Bag-of-Tasks in execution at any given moment. Other workloads are queued by the system and executed in sequence.

The Execution Engine focuses on optimizing two objectives—execution time (performance) and cost. The cost aspect is very important if the resources are rented from cloud service providers which charge for the usage.

In our system the execution involves a three step procedure in which candidate resources are (1) first sampled in order to extract performance metrics related to the application. Using the metrics and a formal model of the system together with a specific resource configuration we can (2) predict the total execution time of the application, without actually running the application on a full configuration. This is achieved by modeling the underlying system and its behavior and using the sampled metrics as input values. Of particular importance is the behavior of the system in the presence of I/O bottlenecks as they are usually the limiting factor when it comes to scaling resources.

In the final step (3), as the resources have been selected by the user, the Execution Engine provisions or acquires the resources, deploys the application and starts the execution.

The resource requirements of the application are not known a priori. They are determined in the sampling phase when a small subset of the bag is executed on each type of available resource.

To account for the I/O limitations, in the makespan prediction phase we employ the queuing network model presented in Fig. 6. They network contains one storage server, where the content is stored, and  $K$  compute resources or nodes rented from the cloud service provider. Each node has associated two rates, compute ( $\mu_i$ —determined during sampling) and maximum download rate ( $\mu_i^D$ ),  $i = 1 \dots K$ . The source server also has an associated service rate ( $\mu_S$ ) which corresponds to its network capacity. In order to predict the performance of the system for a given number and type of nodes, i.e. the makespan, we need to determine the actual

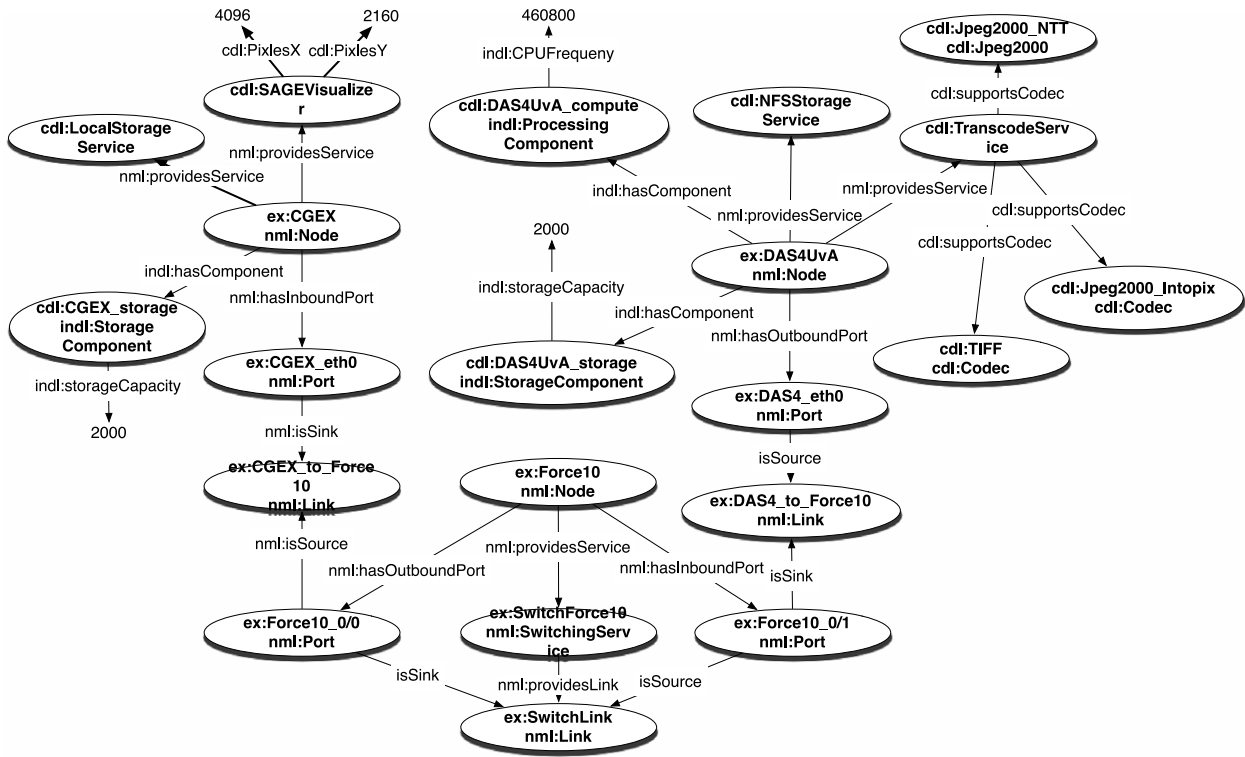


Fig. 2. CDL Description of CineGrid Exchange Amsterdam.

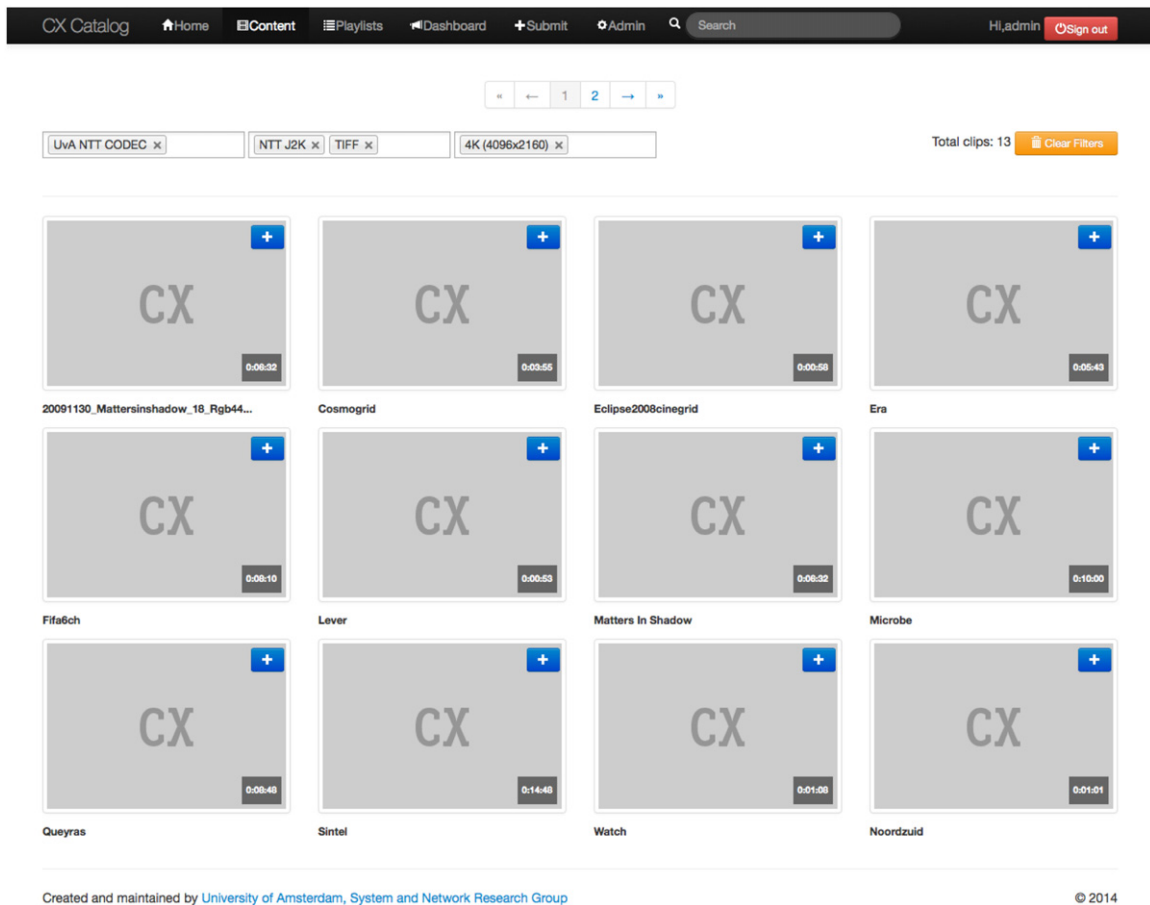


Fig. 3. CGtoolkit Web Portal – Browsing the Catalog – By leveraging the content metadata, users can filter content using criteria like resolution, physical location or available format.

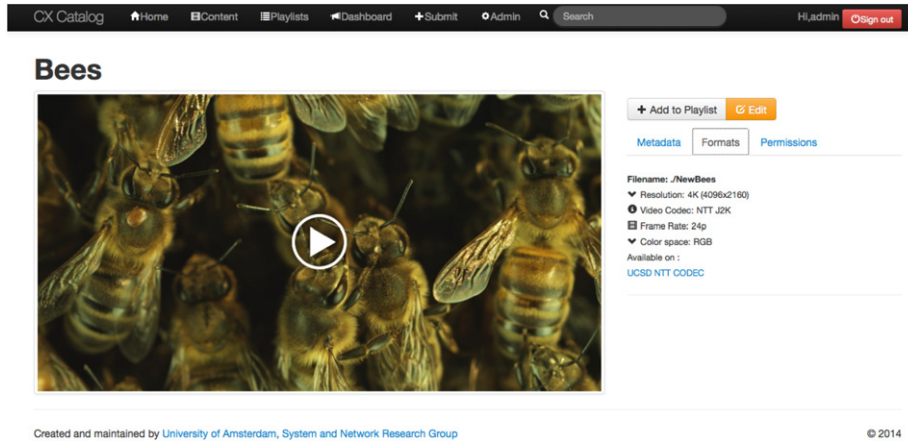


Fig. 4. CGtoolkit Web Portal – Content detail view – Users can inspect the metadata of the assets in detail.

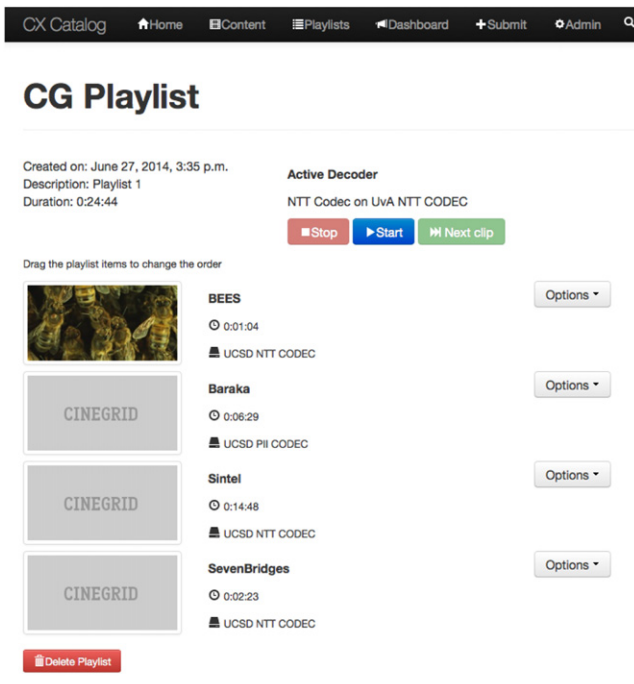


Fig. 5. CGtoolkit Web Portal – Streaming playlist view – Network resources are provisioned dynamically for each item in the playlist.

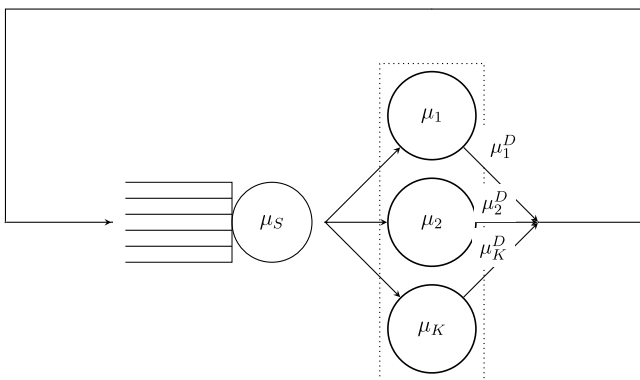


Fig. 6. The queuing model of the execution engine.

download rates for each node, as the network bandwidth will be shared among all the nodes and thus they will not download data at the full rate. While there are methods which can determine

the complete distribution of download rates for each node, they increase in complexity as the number of nodes increases. We have opted for a fast evaluation of the system using the Mean Value Analysis (MVA) [13] method. As the name suggests, this method outputs only the mean values of the system properties, yet it is very fast as it does not have to explore all the states of the system. The high evaluation speed allows the quick exploration of a large number of resource configurations in a short time. Users will only be interested in those resources configurations which perform best with regards to time and cost and our system will output as resource candidates only those configurations. The system evaluates all the possible configurations and presents the user only the relevant solutions. In effect, these resource configurations form the Pareto-optimal set of resource configurations.

The complexity of the evaluation method depends only on the square of size of the configurations, i.e. number and type of nodes. In practice the number of resources used is relatively small but if needed, instead of fully exploring the solution space, heuristics approaches like genetic algorithms or simulated annealing can be used to converge quicker to the optimum. Such heuristics would still use as cost function the MVA evaluation method.

In [14] we present a detailed model and evaluation of this execution prediction method.

The users can build their Bags-Of-Tasks workloads using a graphical interface. A series of image kernels (individual transformations) can be chained together to form an image pipeline. The whole pipeline applied to a single image file consists of a task in the Bag-Of-Tasks. In Fig. 7 we present the web portal view in which users compose the image pipeline. Among the list of kernels which the system is able to execute we highlight: decompressing an image to an uncompressed format, performing color correction on the uncompressed image, encoding it to another format (e.g., JPEG, JPEG2000, TIFF). The image kernels are implemented using generic image processing tools like *imagemagick* [15] and *openjpeg* [16].

#### 4. Evaluation and results

In this section we present the evaluation of the system described in Section 3.

The usability aspect of the web portal has been evaluated during various live demonstrations by both technical experts and regular users. The system has been successfully showcased during events like Supercomputing Conference 2011, CineGrid Day Amsterdam 2012, TERENA Networking Conference 2012, CineGrid Workshop 2012 etc. Two modules of the system, the content metadata repository and the streaming functionality, have also been integrated into the Vroom [17], “an augmented environment

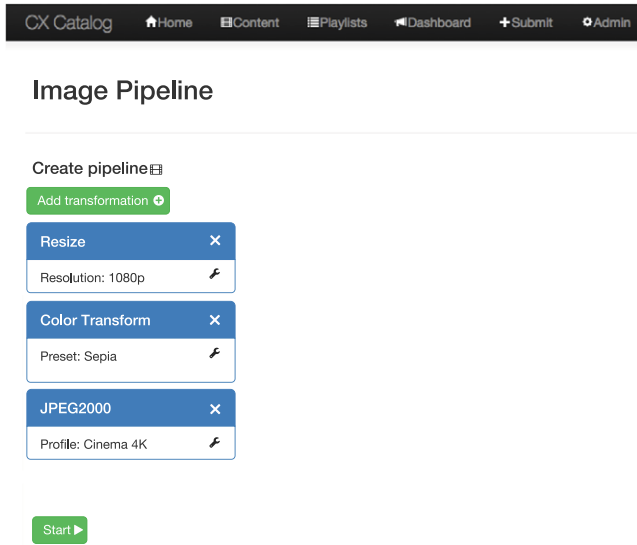


Fig. 7. CGToolkit Web Portal – Image pipeline creation – Users can add, remove, configure and change the order in which transformations are applied to the input images.

for remote collaboration in digital cinema production”. There it is used for playback of remote content using dedicated streaming and decoding hardware.

We have performed various experiments to demonstrate the ability of the system to help users in making optimal resource selection choices. In this section we highlight the situation when cloud resources are used to execute the workload. As we use a public cloud provider and data transfer takes place over the Internet, we cannot adjust the end-to-end network capacity dedicated to the execution. This scenario has the potential to exhibit the scalability issue mentioned in Section 2. A non-expert user would be tempted to acquire as many cloud resources as possible in the hope of achieving the shortest execution time.

We have selected two image pipelines or applications, for the experiments. The first pipeline consists in a compression operation: from TIFF to the JPEG2000 file format, using the *openjpeg* tool configured with the Cinema4K preset. The second pipeline resizes the image to  $1920 \times 1080$  pixels and applies a sharpen filter to the image. In the remainder of this section, we will refer to the pipelines as *openjpeg* and *imagemagick* respectively.

The data was located on one of the *CineGrid* Exchange nodes hosted by the University of Amsterdam and consisted of the first 1500 image frames of the open source movie *Sintel* in the TIFF format. The average file size was 24.3 MB. The measured bandwidth from the Amazon cloud site to the storage node was 700 Mbit/s.

Table 1 presents the characteristics of all the compute resources used in the experiments. The *m1.s*, *m1.m*, *c1.m* names correspond

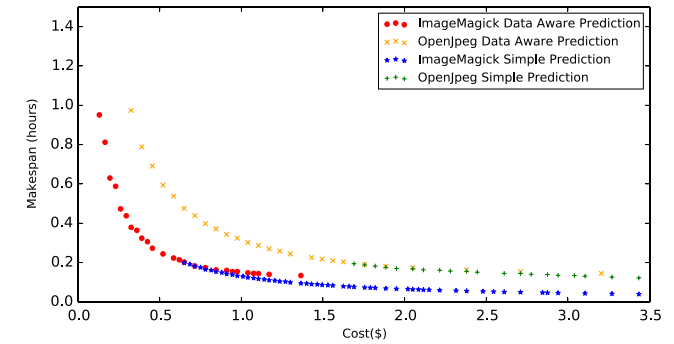
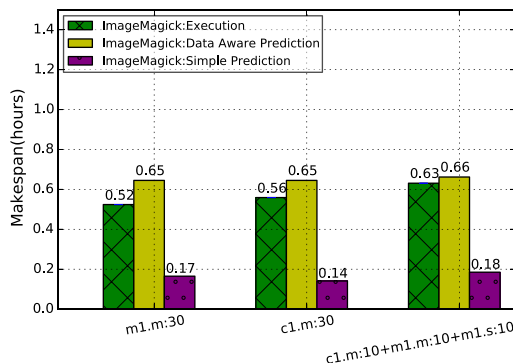


Fig. 9. Pareto set.

Table 1

Compute resource details.

Type	CPUs (ECU)	Memory (GB)	Network	Cost (\$/h)
m1.s	1(1)	1.7	Low	0.047
m1.m	1(2)	3.75	Moderate	0.095
c1.m	2(5)	1.7	Moderate	0.190

to the *m1.small*, *m1.medium* and *c1.medium* instance types offered by Amazon in the Europe, West region.

We simulate the choice of an unexperienced user by executing the image pipelines using configurations which have high number of cloud instances ( $\sim 30$ ). Each configuration is labeled using the types and respective number of instances, using the format: *type:no\_instances[+type:no\_instances[...]]*. For each execution we present the real execution time, the data aware predicted execution time as computed by the execution engine and the simple predicted execution time. Using the execution engine described in [14], we have sampled the performance of each type of cloud instance. The simple predicted execution time is derived by ignoring the limited network resource and is computed assuming that the system performance improves as more instances are added. We observe that there is large gap ( $>50\%$ ) between the simple predicted time and real execution time. The data aware prediction is in all cases very close to the actual measured value ( $<10\%$ ) (see Fig. 8).

Next, we evaluate another feature of the execution engine: the ability to generate accurate Pareto-optimal configurations. The Pareto-sets corresponding to each of the image pipeline is presented in Fig. 9. As in the previous graph, the simple prediction assumes no network contention. We note that the configurations from the Pareto set are those configurations which are able to make the most efficient use of the compute resources. The presence of contention decreases the efficiency of the system and therefore a Pareto-optimal configuration would be, as much as possible, contention free. As the set of Pareto-optimal configurations is quite

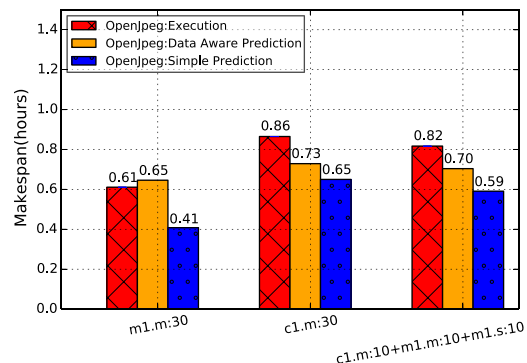


Fig. 8. Execution results for large configurations (30 cloud instances).

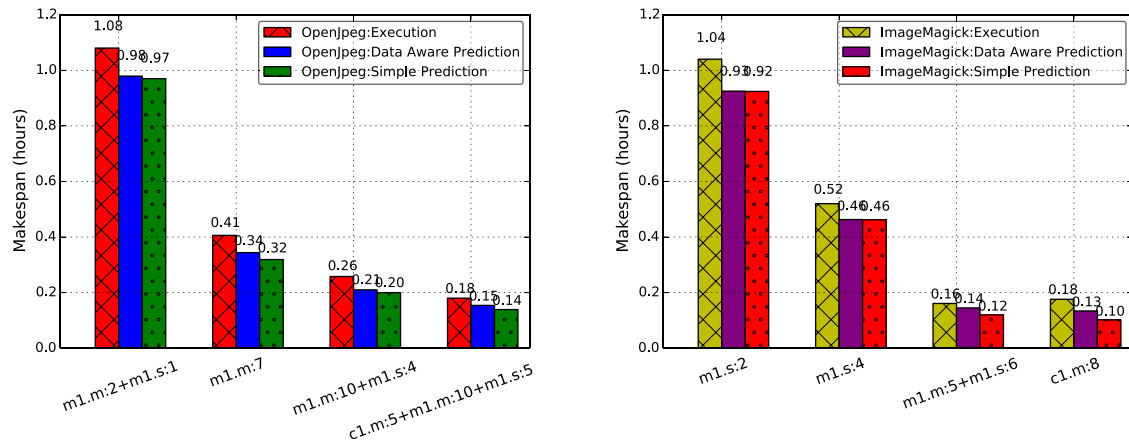


Fig. 10. Measured, 'data aware' and 'simple' predicted execution times for configurations from the Pareto front.

large (more than 30 different configurations for each application type), we have selected four configurations from the set, for each image pipeline. The configurations selected include the cheapest and most expensive configurations predicted using the data aware method. The other two configurations are selected such that they divide the cost interval between the cheapest and most expensive configurations in equal parts. We have made this choice in order to have a good coverage of the Pareto set. The results of the execution are shown in Fig. 10. In this case the prediction method has been more optimistic than the real execution, yet in all cases the misprediction is quite small ( $< 10\%$ ).

The experiments above demonstrate the ability of the system to present to the unexperienced user correct Pareto optimal resource configurations.

## 5. Related work

As our work integrates various components we present in this section related work to both integrated systems like ours and to work which is related to various subcomponents of our system.

### 5.1. Resource selection, runtime prediction and scheduling

The simplest resource selection strategy which can be employed is the one in which the user knows a priori the application's requirements and the behavior of the available resources. In this case the user manually selects which resources should be used for execution. While this is simple, there is no guarantee on the system's performance and the only way to improve this strategy is to repeatedly adjust the selection of resources in the hope of achieving better performance. Another downside of this is that the user needs to understand both the performance characteristics of the workload and of the resources. This is an unreasonable expectation for a user-friendly system, which assumes minimal user-side input.

We turn our attention to more advanced methods which require less detailed knowledge from the user's side. The user's requirements are usually expressed through objective functions which need to be either minimized, e.g. cost or energy consumption, or maximized, e.g. performance or throughput. One basic requirement to achieve this is the ability that given a resource configuration, the system should be able to predict as closely as possible the runtime of the application.

Most approaches use historical data to match the current application with similar applications executed in the past. There are numerous examples in literature, from simple regression to more advanced machine learning techniques [18–20]. In all cases the efficiency is highly dependent on the training set available prior to

the prediction phase. In [21] the authors use a similar prediction method, in which the Bag-of-Tasks is sampled at runtime. Cost and makespan are considered as objectives. However, the authors ignore the network component and assume that the I/O performance is constant regardless of the number of compute resources used. In [22] the workload is also sampled and then partitioned in sub-applications which have similar *Communication-To-Computation* ratios. Based on this partitioning the authors employ then a heuristic to assign jobs to the available resources. The objective of the heuristic is to maximize the throughput of the system. The resources have no associated cost and the evaluation of the heuristic is done only through simulations. Our model uses the steady state model of job execution [23] in which initialization and cleanup phases are ignored. This relaxes the scheduling problem and allows us to focus on the high throughput phase of the execution.

### 5.2. Pareto optimality

The next important requirement is generating *relevant* options for the user, given that we can estimate with good accuracy the runtime of their applications on any configuration. We focus on the Pareto optimal of resource configurations. Typically users target multiple objectives which potentially can influence each other, e.g. cost and performance, so a configuration cannot excel for both metrics. The Pareto set provides those configurations which are *non-dominated*. In a non-dominated configuration one objective cannot be improved without worsening another [24]. This gives a wide range of optimal choices in terms of trade-offs available to the user. In [25] the authors explore different scheduling strategies for Bags-of-Tasks on unreliable cloud resources in order to provide Pareto-optimal execution options. In our work we assume that the pool of resources is reliable. Following a slightly different approach the authors in [26] also provide Pareto-optimal scheduling on hybrid infrastructures, private and public clouds. However, the workloads which the authors target do not include any resource contention like network or storage.

### 5.3. Networked media processing

Media processing and delivery systems using dynamic infrastructures have become quite popular in recent years. Most systems focus on the ability of cloud resources to scale with the user demand. While there are works which target digital cinema and very high quality media, for example FOGO player [27] and UltraGrid [28], they are limited to streaming and visualization. A close approach to ours is the architecture described in [29] which describes how high speed optical networks could be used in the

digital cinema domain. Yet, this work only presents a theoretical framework for managing multiple high bandwidth video streams on a network topology where dedicated paths can be created dynamically. Other aspects such as content or service selection or distributed processing are not addressed.

## 6. Future work

One of the current assumptions of the *CGtoolkit* is that the workload consists of independent tasks which can be executed in any order. This assumption does not hold for more complex workloads or for processing data streams in which input becomes available in a specific order. We intend to explore the scheduling policy used for distributing the tasks to the workers. By having more information about the workload, i.e. per task resource requirements, we could change the scheduling policy to make more informed mappings between resource types and tasks. In [22] the authors suggest that for workloads which are ‘balanced’ i.e., are not either compute dominated or data transfer dominated, the demand driven scheduling policy is not the most efficient. However even with advanced scheduling policies the problem of efficiently scaling each phase remains as the data contention aspect would be still present. Finally, we assume that the approach we took is not limited only to media workloads, and hence we plan to experiment with different applications which exhibit similar data intensive behavior and which map to the same execution strategy.

## 7. Conclusion

In this paper we have presented *CGtoolkit*, a user-centric video processing and delivery system for *CineGrid* workloads in heterogeneous environments. The system is used in the *CineGrid* collaboration by a diverse range of users, experts and non-experts. The ability to efficiently use complex computing resources increases productivity in the media world. In addition, it stimulates collaboration and experimentation of new technological developments in the network and high performance computing work.

## Acknowledgment

We would like to thank the *CineGrid* NL project for sponsoring this work.

## References

- [1] S. Liu, J.P. Schulze, L. Herr, J.D. Weekley, B. Zhu, N.V. Osdol, D. Plepys, M. Wan, *CineGrid Exchange: A workflow-based peta-scale distributed storage platform on a high-speed network*, *Future Gener. Comput. Syst.* 27 (7) (2011) 966–976.
- [2] OSDC—Open Science Data Cloud. <https://www.opensciencedatacloud.org/> (accessed: 01.03.14).
- [3] ExoGENI. <http://www.exogeni.net/> (accessed: 01.03.14).
- [4] Amazon EC2—Amazon Elastic Compute Cloud. <https://aws.amazon.com/ec2/> (accessed: 14.03.01).
- [5] Cinegrid, <http://cinegrid.org/>.
- [6] Global lambda integrated facility. <http://www.glif.is>.
- [7] Sintel: Open source movie. <http://sintel.org/>.
- [8] A. Iosup, O. Sonmez, S. Anoop, D. Epema, The performance of bags-of-tasks in large-scale distributed systems, in: *Proceedings of the 17th International Symposium on High Performance Distributed Computing*, ACM, 2008, pp. 97–108.
- [9] M. Ghijzen, J.V.D. Ham, P. Grosso, C.D. Laat, Towards an infrastructure description language for modeling computing infrastructures, in: *10th IEEE International Symposium on Parallel and Distributed Processing with Applications*, ISPA 2012, 10–13 July 2012.
- [10] R. Koning, P. Grosso, C. Laat, Using ontologies for resource description in the *CineGrid Exchange*, *Future Gener. Comput. Syst.* (2010) 1–15. URL: <http://linkinghub.elsevier.com/retrieve/pii/S0167739X10002475>.
- [11] OpenNSA. <https://github.com/jeroenh/OpenNSA> (accessed: 27.01.14).
- [12] A. Benoit, L. Marchal, J.-F. Pineau, Y. Robert, F. Vivien, Scheduling concurrent bag-of-tasks applications on heterogeneous platforms, *IEEE Trans. Comput.* 59 (2) (2010) 202–217.
- [13] S.S. Lavenberg, *Computer Performance Modeling Handbook*, Academic Press, Inc., Orlando, FL, USA, 1983.
- [14] C. Dumitru, A.-M. Oprescu, M. Živković, R. van der Mei, P. Grosso, C. de Laat, A queueing theory approach to pareto optimal bags-of-tasks scheduling on clouds, in: *Euro-Par 2014 Parallel Processing*, Springer, 2014, pp. 162–173.
- [15] ImageMagick: Convert, Edit, Or Compose Bitmap Images. <http://www.imagemagick.org/> (accessed: 27.01.14).
- [16] OpenJPEG—JPEG2000 Codec. <http://www.openjpeg.org/> (accessed: 27.01.14).
- [17] T. Margolis, T. Cornish, Vroom: designing an augmented environment for remote collaboration in digital cinema production (2013) 86490F–86490F–11 <http://dx.doi.org/10.1117/12.2008587>.
- [18] A. Matsunaga, J.A. Fortes, On the use of machine learning to predict the time and resources consumed by applications, in: *Proceedings of the 2010 10th IEEE/ACM International Conference on Cluster, Cloud and Grid Computing*, IEEE Computer Society, 2010, pp. 495–504.
- [19] K. Sembiring, A. Beyer, Dynamic resource allocation for cloud-based media processing, in: *Proceeding of the 23rd ACM Workshop on Network and Operating Systems Support for Digital Audio and Video*, ACM, 2013, pp. 49–54.
- [20] O. Sonmez, N. Yigitbasi, A. Iosup, D. Epema, Trace-based evaluation of job runtime and queue wait time predictions in grids, in: *Proceedings of the 18th ACM International Symposium on High Performance Distributed Computing*, ACM, 2009, pp. 111–120.
- [21] A.-M. Oprescu, T. Kielmann, H. Leahu, Budget estimation and control for bag-of-tasks scheduling in clouds, *Parallel Process. Lett.* 21 (02) (2011) 219–243.
- [22] H. Casanova, M. Gallet, F. Vivien, Non-clairvoyant scheduling of multiple bag-of-tasks applications, in: *Euro-Par 2010-Parallel Processing*, Springer, 2010, pp. 168–179.
- [23] O. Beaumont, A. Legrand, L. Marchal, Y. Robert, Steady-state scheduling on heterogeneous clusters, *Internat. J. Found. Comput. Sci.* 16 (02) (2005) 163–194.
- [24] A. Messac, A. Ismail-Yahaya, C.A. Mattson, The normalized normal constraint method for generating the Pareto frontier, *Struct. Multidiscip. Optim.* 25 (2) (2003) 86–98.
- [25] O.A. Ben-Yehuda, A. Schuster, A. Sharov, M. Silberstein, A. Iosup, Expert: Pareto-efficient task replication on grids and a cloud, in: *Parallel & Distributed Processing Symposium (IPDPS)*, 2012 IEEE 26th International, IEEE, 2012, pp. 167–178.
- [26] M. HoseinyFarahabady, Y. Lee, A. Zomaya, Pareto-optimal cloud bursting.
- [27] L.A. Júnior, R. Gomes, M.S. Neto, A. Duarte, R. Costa, G.L. de Souza Filho, A software-based solution for distributing and displaying 3D UHD films, *IEEE MultiMedia* 20 (1) (2013) 60–68.
- [28] P. Holub, L. Matyska, M. Liška, L. Hejtmánek, J. Denmark, T. Rebok, A. Hutanu, R. Paruchuri, J. Radil, E. Hladká, High-definition multimedia for multiparty low-latency interactive communication, *Future Gener. Comput. Syst.* 22 (8) (2006) 856–861.
- [29] O.-D. Ntounou, D. Simeonidou, D.K. Hunter, Cloud-based architecture for deploying ultra-high-definition media over intelligent optical networks, in: *2012 16th International Conference on Optical Network Design and Modeling (ONDM)*, IEEE, 2012, pp. 1–6.



**Cosmin Dumitru** is a Ph.D. student in the SNE group at the UvA. In 2010 he received his master's degree in system and network engineering at the same university with the 'Cum Laude' distinction. Cosmin's research is focused on high-performance networking and distribution and processing of high definition media over optical networks. Currently he is involved in the *Cinegrid* project.



**Paola Grosso** is assistant professor in the SNE group at the UvA. Her research interests are in the area of Future Internet architectures and the underlying models that can support their operations. In particular she studies how hardware and network infrastructures can contribute to the creation of 'greener' and sustainable ICT services.



**Cees de Laat** is chair of the System and Network Engineering research section at the University of Amsterdam. Research in the section includes optical/switched data-transport in TeraScale eScience applications, Semantic web to describe e-Science infrastructure, distributed authorization architectures, virtualized cyber-infrastructure and Security& Privacy. <http://delaat.net/>.