From scripted HPC-based NGS pipelines to workflows on the cloud

Jacek Cała*, Yaobo Xu†, Eldarina Azfar Wijaya* and Paolo Missier*
*School of Computing Science
Newcastle University, Newcastle upon Tyne, UK
†Institute of Genetic Medicine
Newcastle University, Newcastle upon Tyne, UK
Email: {firstname.lastname@newcastle.ac.uk}

***Abstract*—In this paper we describe our initial experiences in the *Cloud-e-Genome* project with moving the whole exome sequencing pipeline from the scripted HPC-based solution to a workflow enactment system running in the cloud. We discuss shortcomings of the existing approach based on scripts and list benefits that a workflow-based solution can provide. Despite the effort it involved to wrap all required tools in the form of workflow blocks and the restrictions of the dataflow model used to represent workflows we expect the migration to significantly improve the current status of the pipeline. Our target is to enable flexibility, traceability and reproducibility of the solution, so that it can better fit the evolution of tools, data and pipeline itself and allow us to run it at national scale. This work will become foundation for the more complete system that includes variant filtering and interpretation for the diagnostic purposes.**

I. Introduction

The cost of genomic sequencing has been decreasing more than linearly over the past few years. Figures from 2010 show the number of base pairs (bp) that can be sequenced per $ unit, doubling every five months [1]. In the UK, the cost of sequencing a single patient sample is currently just under $1.5K and decreasing.

Combined with the cost-effective availability of scalable cloud computing infrastructure, the economics of genome sequencing are dramatically changing the traditional perception of genetic testing as a method of last resort for the diagnosis of unfamiliar and rare genetic diseases. In fact, Next Generation whole-exome sequence (WES) and, soon, whole-genome sequence data (WGS) for disease-gene identification are predicted to become routine diagnostic tools in clinical practice within a few years, especially for the diagnosis of rare diseases. WES-based diagnosis involve sequencing the coding portions of the patient's genome (the exome), and then extracting information from it about potentially pathogenic mutations (gene variants) using a variety of published algorithms and tools, both established and experimental.

In this paper we describe preliminary findings from the *Cloud-e-Genome* project, a collaboration between the Institute of Genetic Medicine and the School of Computing Science at Newcastle University, funded by the Biomedical Research Centre in the UK.[1] The project's overall goal is to facilitate the adoption of systematic and reliable genetic testing in clinical practice, at population scale, within the UK. This vision of routine population-scale genetic testing presents a real scalability challenge. Genetic diseases affect 6–8% of the population in the UK, or about 5 million people who could benefit from genetic testing. At the same time, exons only account for about 1% of the human genome [2], and that is broadly acknowledged to be insufficient for a robust diagnosis in all cases. The ability to routinely sequence the entire genome (WGS) will soon bring the remaining 99% into the picture.

To give an idea of computational cost, it currently takes about 18 hours of computing time, on our local HPC cluster, to obtain a list of variants (SNVs) from one sample. This time increases when samples of a patients relative(s) are also sequenced, to improve diagnostic accuracy. It further increases when one considers reprocessing old sequences, following evolution in any of the many algorithms and tools involved in variant calling. If we add to this the expected 100 fold scale up in data size and processing times for the upcoming third generation sequencing technology, it becomes clear that a highly scalable, yet economically affordable storage and computing infrastructure is required.

A. Flexibility, traceability and reproducibility requirements

This evolving scenario suggests two key additional requirements for any NGS processing system, in addition to scalability.

a) Flexibility and extensibility: Firstly, the system must be able to track technology evolution by rapidly adapting an existing pipeline, i.e., by reconfiguring tools (changing their configuration parameters to explore different options), rapidly adding new tools, tracking their evolution across versions. As newer versions of some of the algorithms become available, old sequences need to be re-processed, as the set of variants called by the new version is very likely to be different from the original one. In fact, even slightly different configurations of tools within a single pipeline are known to produce different results.

b) Traceability: Secondly, associated with the variability in variant lists produced at different times, is also the need to explain the differences amongst those sets of variants, in terms of differences between the pipelines that produced them. In turn, this brings about the requirement to produce a detailed trace of pipeline execution, and thus to record and store the

[1]Partial funding for the project comes from NIHR (National Institute for Health and Research) as well as from a grant from Microsoft Research under their *Windows Azure for Research Award*.

provenance of each and every set of variants, in a way that makes it amenable to explaining differences observed amongst the output datasets.

Importantly, the need to systematically track the variant calling process extends to downstream phases of WES-based, and in the near future, WGS-based testing, namely variant interpretation for diagnostic purposes. One of the promises of WES/WGS is the ability to test multiple disease hypotheses regarding common and, perhaps more interestingly, rare genetic diseases. Given one or more phenotypes that are derived from the patient's clinical observations, the task of variant interpretation in a clinical setting involves isolating those variants (generally very few) that have a high likelihood of being pathogenic for the specific patient, and are located on genes that are known to be associated with the phenotypes. Techniques for variant filtering and selection are the subject of active and intense research (see [**?**], [**?**] for a list of references). The main concern, for WES to become usable by clinicians, is to guarantee a very low false positives rate (i.e., the chance to erroneous deliver a disease diagnosis to the patient). Currently, variant filtering is still a semi-automated, knowledge-intense task which is broadly based on heuristic rules, and is partly in the hands of human experts. The ability to track the filtering steps is therefore critical to explaining the decision process that ultimately led to a diagnosis. Thus, once again making the entire process provenance-aware is a key requirement for our project.

c) Reproducibility: Related to traceability, the third major requirement for *Cloud-e-Genome* is support for reproducibility. Not only is it necessary to provide evidence for the outcome of a complex process; the process must also be repeatable over time, and furthermore, it should be possible to compare results obtained at different times, possibly using different versions of the tools involved.

B. From scripted pipelines on HPC to workflows on the cloud

In the rest of the paper we describe our technical approach to addressing the requirements set out above, focusing solely on phase I of *Cloud-e-Genome*, namely variant calling.

WES is currently in use within our local research setting (i.e., not for general clinical use) at the Institute for Genetic Medicine. The pipeline is illustrated in Fig. 1 and includes typical NGS processing steps [3], i.e., alignment (BWA), cleaning (Picard), sequence recalibration, filtering, variant calling and recalibration (GATK), coverage analysis (bedTools), and annotation (Annovar). The implementation consists of a number of shell scripts that coordinate the sequential execution of each of the tools, and is deployed on a dedicated cluster running the Alces cluster management software.

Our approach involves a double porting effort. Firstly, the pipeline has been re-coded using our homegrown Workflow Management System (WfMS), e-Science Central (e-SC). e-SC has been under active development for over 5 years within the School of Computing Science, and in past projects has proven itself as a viable programming model and execution environment on multiple cloud infrastructures, including Azure [4], [5]. It also provides provenance recording and serialization using the interoperable Open Provenance Model [6].

One of the main goals of workflow technology is to empower scientists with no specific programming background to take control of the design of their data processing pipelines. Workflow models achieve this goal by reducing the programming effort by increasing the level of abstraction, so that it becomes an intuitive assembly of pre-defined components. This involves separating two sets of tasks, namely (i) developing new pipeline components and making them available to designers, and (ii) creating new workflows (e.g. an NGS pipeline) using those components. The former requires competence both on the tools and their interfaces, as well as in general programming. Workflow design, on the other hand, can be tackled by domain experts (i.e., geneticists) with only a modest amount of specific training.

With this in mind, re-coding the pipeline involved two types of initial development effort. Firstly, the tools that make up the existing pipeline must be turned into workflow blocks. Secondly, an initial version of the pipeline, modelled on the existing one, must be reconstructed using the available blocks. As discussed in detail in Sec. III-A, this is itself a complex task, as it entails moving from a control-based implementation (scripts) to a dataflow model (the model is briefly described in Sec. III-C). Once this initial setup is complete, however, the resulting workflow provides domain experts with a platform that they can use to experiment on their own by introducing minor variations (e.g. changing the values of some tools' configuration parameters). Incremental pipeline evolution may still involve deploying new blocks, however.

Another element of our porting effort involves deploying on a public IaaS cloud (Microsoft Azure). We contend that an architecture based on cloud-deployed workflow meets the requirements of scalability and flexibility in a way that is cost-effective and does not compromise performance, thus opening the way to operational deployment for clinical use. Furthermore, provenance recording available in e-SC enables detailed traceability of process execution. These traces can be used not only as a form of evidence on how the results were obtained, but also, in some cases, to describe the effect of minor variations between two workflow specifications, i.e. either in the structure or in the versions of some of the tasks [7].

II. Related Work

Our architecture is aligned with Stein's vision of a "genome informatics ecosystem" centred around a cloud computing infrastructure, where most genome datasets are stored centrally on virtual servers, and computing resources are allocated elastically to cope with the requirements of increasingly complex data processing pipelines [1]. Indeed, cloud computing has recently and repeatedly been advocated as a viable infrastructure and economic model to support scalable data-intensive computing, and exemplars of cloud-based data processing architectures for genomics are beginning to emerge. Recent examples include the suite of cloud-based bioinformatics tools described in [8] and the Advanced Sequence Automated Pipeline (ASAP) framework [9]. Both these solutions offer little flexibility for re-coding and experimenting with workflow variations, however. The dependency on external tools that are not under the control of the "workflow" owner also suggest limitations in the reproducibility of the analysis.

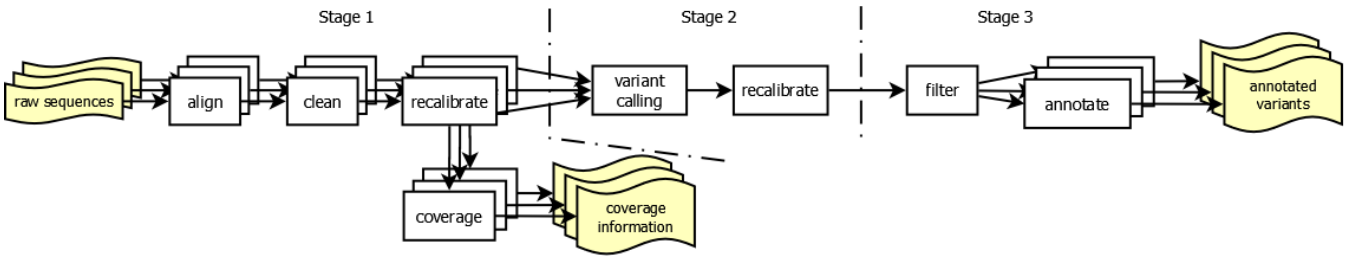

Fig. 1. Original NGS processing pipeline

Galaxy is a broadly adopted and successful platform for genomics studies [10]. Galaxy provides ease of programming, reproducibility by way of an eco-system of dedicated tools, provenance recording, and recently, cloud deployment of the pipelines. This is achieved by means of CloudMan [11], [12], a cloud manager for the provisioning and control of the required data analysis environment on popular cloud infastructures. Galaxy also integrates functionalities from the popular Taverna workflow manager [13] with cloud computing support (Tavaxy [14]).

Arguably, Galaxy comes close to satisfying the requirements we set above. Used in conjunction with Globus Online services, Condor and Amazon EC2, its effectiveness was demonstrated in several use cases including ChIP-, RNA- and Exome-sequencing [15]. The solution shows, however, that to run a complete NGS pipeline in the cloud Galaxy needs some additional support tools. In contrast, our goal is to offer an integrated solution that comprises all software stack needed to run scientific analyses and which requires nothing more than the user's input data. Furthermore, in favour of e-SC is also its ability to distribute a workflow computation across cloud cluster nodes in a user-controlled manner. Briefly, workflows may contain special tasks which are used to start sub-workflows. Each sub-workflow can then be run on a separate worfklow engine, and multiple engines can be deployed on distinct nodes. Thus, workflow designers explicitly control the degree of available parallelism by specifying fork-able sub-workflows at suitable points within the main workflow. Technical arguments aside, as a home-grown system e-SC gives us complete control over the evolution of the workflow model, its technical features, and its future deployment model over a variety of cloud infrastructures.

A different deployment model involves replicating HPC cluster functionalities on a public cloud, as the success of the UPPNEX HPC cluster infrastructure in Sweden shows [16]. The elasticHPC software package and library, designed to simplify the use of high performance cloud computing resources for bioinformatics applications, is another example [17]. Provided these models can be successfully combined with a workflow-based implementation of the pipelines, we may consider this option as an alternative in the future.

Ultimately, the adoption of e-SC as a workflow platform for e-science has been proven viable by the experience of the recent project *Cloud4Science* [4], where programmable workflows have been used to orchestrate a selected set of selected bioinformatics tools. These tools are designed specifically to support NGS pipelines in the cloud, with the goal to reduce technical skills requirements on users.

Finally, the potential role of MapReduce implementations of popular tools in genomics is relevant in this context. One strand of tools, eg Cloudgene [18], are aimed at simplifying the execution of MapReduce programs for Bioinformatics through a user-friendly, web-based interface. More directly relevant to an approach based on programmable workflow is the availability of specific tools on top of open source MapReduce implementations, such as Hadoop. A number of such tools are listed in [19], and the popular Genomics Analysis Toolkit (GATK), which is used in our *Cloud-e-Genome* pipeline, is one example [20].

It is easy to imagine how the e-SC deployment model may work alongside workflow blocks that are MapReduce jobs, as they may share the same cloud infrastructure. Specifically, e-SC can distribute sub-workflow computation across multiple cloud-based workflow engines, as mentioned above, while individual blocks are executed on Hadoop cloud nodes. The appeal of this configuration is not proven, however, and this option is left for the future investigation.

III. Technical approach

We now describe the shortcomings of the current pipeline that triggered the migration effort, and the technical challenges associated with it.

A. Shortcomings of legacy pipelines

The migration effort was triggered by a combination of technical and organizational considerations, some of which are arguably common to most "legacy" code. In essence, pipeline development and maintenance requires intimate knowledge of the pipeline, which consists of a collection of low-level, bash scripts, with no abstraction on top of the raw tools command line interface. The developers' knowledge must extend not only to each of the tools configuration, but also to the available deployment options. For instance, job submission to the local HPC cluster requires the explicit allocation of the desired number of nodes and cores within the nodes. This configuration is specific to the tools invoked by the scripts and to the cluster itself, making the entire implementation hardly portable. Maintaining the pipeline is also complicated, in sharp contrast to a rapidly changing NGS tools landscape. For instance inter-task dependencies, and thus effectively the structure of the pipeline, are hidden in the code. This includes knowledge of physical file locations and how files are shared across steps of the pipeline. Also, the cluster provides no isolation. Interference from other cluster users, in the form of apparently minor issues such as saturated scratch shared disk space, causes long-running executions to fail arbitrarily.

In addition to making the code fragile, such complexity dangerously centralizes control, requiring a highly skilled developer role. At the same time, limitations in the developers' technology skills translates into a fragmented pipeline which is not as fully automated as it could be. This happens for instance with handling errors of the tools, which if done incorrectly leads to the need for manual inspection whether the produced result is correct and can become input for the subsequent step.

Ultimately, as it is often the case with low level e-science tools, these limitations get in the way of producing valid sets of variants while tracking technology evolution.

B. The e-SC WfMS

e-SC is a cloud-based data processing system that supports both Software and Platform as a Service (SaaS/PaaS) deployment models for scientific data management, analysis and collaboration. It is a portable system which can be deployed on both private and public clouds (e.g. Eucalyptus, Amazon AWS and Microsoft Windows Azure, respectively). The SaaS interface allows scientists to upload data, edit and run workflows, and share results in the cloud using only a web browser. It is underpinned by a scalable cloud platform consisting of a set of components designed to support the needs of scientists. More advanced users with software development skills in common languages (e.g. Java, R) may use the system in PaaS mode, by uploading their own analysis services into the system and making these available to others in the form of ready-to-use workflow tasks. A REST-based API is also provided so that external applications can leverage the platform's functionality, making it easier to build scalable, secure cloud based applications [21].

e-SC has been used in a number of scientific projects such as spectral data visualisation, medical data capture and analysis, and chemical property prediction [22]. As mentioned earlier, it also featured as the WfMS of choice (in server-only mode) to run simple NGS analyses in the Cloud4Science project [4]. Such instance demonstrated the appeal of providing a higher abstraction level than scripts, as well as the capability to hide the intricacies of workflow design from end users.

Importantly, e-Science Central supports enactment of dataflows as opposed to control-flows. Briefly, the control-flow model defines the order of operations to be executed and allows workflows to include control interactions such as loops and conditionals. It usually also includes some form of common data store so that the operations can manipulate shared variables and data. Conversely, the dataflow model remains very simple and defines only services used and the data dependencies between them [23]. In dataflows there is no global data shared between services and the flow of execution and data is explicitly created by the graph of connections between services. The major advantage of dataflows is the simplicity of the workflow structure which makes them easier to understand by non-expert users. They are also a very good option for visual representation.

C. Migration challenges and associated effort

Considering all the disadvantages of the script-based approach discussed above and potential advantages that e-SC workflows can offer, we decided to migrate from scripts to workflows and from our local HPC cluster to the cloud. The former should result in better pipeline design and easier provenance tracking, whereas the latter should offer us better flexibility in resource access.

The process of migration, although a seemingly simple task, involved substantial effort. It was mainly related to the number of tools and reference data we had to embed as e-SC blocks and libraries and converting the existing script-based pipeline into dataflows. Moreover, it involved extending e-SC to match specific requirements of the tools (e.g. the need to efficiently upload large sequence files), testing partial and complete solution and, finally, the management of the cloud service.

The first step in migrating the pipeline was to wrap existing NGS tools and reference data, like bwa sequence aligner and GATK processing tools, into e-SC blocks and libraries. This process resulted in building 25 blocks and 13 libraries specific to NGS. If the need for blocks is clear, they are used to design workflows, building e-SC libraries provides two benefits. It gives better efficiency in running the tools. The libraries are installed and cached by the workflow engine only once. Moreover, it is a way to ensure reproducibility because we do not need to rely on external services to provide us human reference genome and all other referenced data but we can store and version them under e-SC control.

With all tools ready, the next step was to adopt existing script-based pipeline so it could be run as an e-SC workflow. This required a few changes to the overall pipeline architecture. Firstly, being unable to express control interactions within a dataflow we had to rewrite the loops in the scripts into equivalent constructs. Secondly, we wanted to fully automate the pipeline so no manual intervention was required unless necessary, e.g. unless errors occur.

Figure 1 shows the high-level picture of the WES pipeline as coded in the shell scripts. It closely resembles the best practices defined by the Broad Institute[2] and adds only calculating sample coverage data and variant annotation (also including in house annotation based on a locally available database). The pipeline involves three key stages: (1) preparation of the raw sequences for variant discovery, (2) variant calling and recalibration, (3) variant filtering and annotation. Stages (1) and (3) were run in a loop so that all tools involved were executed for each sample separately. Stage (2) was run only once for all input samples.

Loops were used during stage (1) and (3) to iterate over the number of samples that the pipeline was configured to process. For each sample, the script coordinating stage (1) invokes a set of jobs like sequence alignment, cleaning and recalibration. To model the same process as a dataflow we exploited the fact that blocks can transmit a list of data elements through their ports and also that certain e-SC blocks can start a number of sub-workflow invocations based on the length of an input data list. This creates a clear pattern in which an initial block generates a list of data samples to process and the following block starts a sub-workflow (the loop body) for each element in the list.

[2]http://www.broadinstitute.org/gatk/guide/best-practices

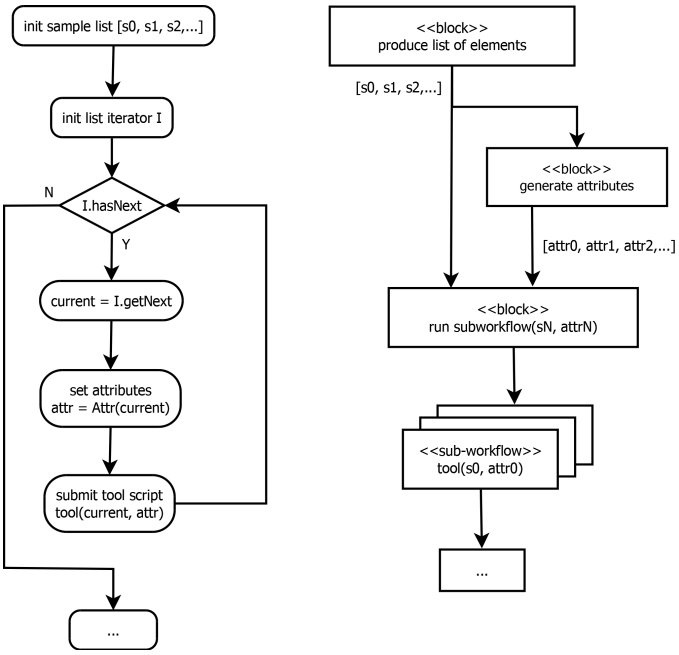

Fig. 2. The general pattern how loops in script can be converted into a dataflow.

However, the loop construct in a script allows also for data manipulation and state variables to be used within the loop body. Our pipeline scripts used a few state variables to rewrite read group information for the input sequences. As variables are not allowed in e-SC dataflows directly, to rewrite read groups we had to implement a custom block that can generate relevant information based on input data. In fact, this custom block implements a loop that mimics exactly the loop from the script. Instead of calling the loop body, however, it outputs a data list that is used by a subsequent, generic block to call the sub-workflows that realize the loop body. Figure 2 shows a pattern which converts a loop from a script into an equivalent dataflow. The pattern can be used only for loops that do not include self-references and so which are not inherently sequential like e.g. calculating Fibonacci sequence.

To unwind the loop in stage (3) we used the same pattern. In this case, though, the loop included simple string manipulation and so we were able to avoid implementing a custom loop block and used existing blocks provided by e-SC. This was very useful because such custom blocks are very specific to a particular case and are hardly reused in any other scenarios.

IV. Discussion

Moving the NGS pipeline to the cloud was based on the assumption that the cloud is able to give us more flexibility in scaling resources up and down. To achieve other benefits like reproducibily and extensibility we decided to adopt the workflow-based approach discussed earlier. Clearly, however, there are other ways in which this migration cloud be done.

One of the easiest ways would be to run the SGE environment in the cloud and transfer data and scripts verbatim; this could be facilitated by solutions such as StarCluster[3] or CloudMan [12]. On the one hand, it would save us substantial effort in wrapping tools into workflow blocks and designing workflows. On the other hand, we would suffer from most of the issues and problems that have been observed in the cluster such as: low-level abstraction of scripts, unclear data dependencies, no metadata management, no provenance. If we add to this significantly lower performance of the cloud than HPC (c.f. [24], [25]), the only benefit of running SGE in the cloud would be flexibility in resource scaling.

Another approach to consider, as discussed earlier in section II, could involve using MapReduce to implement the pipeline. MapReduce fits, however, the level of NGS tool implementation (see e.g. [20]) and thus would still require a higher abstraction level to match our requirements.

Our approach was to use a workflow framework to implement the pipeline and move it to the cloud, so that we could eliminate the most annoying issues inherent to using shell scripts and add some of the attractive features offered by e-SC. The main advantages of this migration are:

- resource scaling,
- version control,
- data provenance,
- uniform tool interface,
- easier sharing of pipelines.

Firstly, it is easy to manage cloud resources in the pool and grow or shrink the cluster according to the actual needs. Importantly, it is also easier to grow the data storage space. In Azure the current limit for a storage account is 200 TB[4] which is substantially more than 60 TB of working space available in our HPC cluster.

Secondly, in e-Science Central changes to data, workflows and workflow blocks are automatically tracked and every change generates a new version of the object. This gives a detailed insight into which tool, data and workflow version was used to produce particular result. Although similar level of version control could be achieved by means of source version control systems such as git and subversion, the advantage of versioning provided by e-SC is that it happens automatically. It does not require any user intervention until the user is interested in other than the latest version of an object. For less experienced users it greatly minimizes the learning curve of managing versions.

Thirdly, together with version information, e-SC keeps track of data provenance, which enables users to investigate also the configuration of workflows and blocks that accessed or produced data objects. As mentioned earlier, we find this feature very important in achieving traceability of results.

A valuable side effect of wrapping NGS tools into e-SC workflow blocks is that the blocks offer a uniform, clearly defined user interface; they have input and output data ports and input properties. This, again, lowers the learning curve for less experienced users but is also convenient for experts. Users can simply focus on the actual function of a tool and required

[3]http://star.mit.edu

[4]See "Windows Azure Storage Scalability and Performance Targets" at http://msdn.microsoft.com/en-us/library/windowsazure/dn249410.aspx

input rather than on remembering what execution environment it needs and how to set desirable input. It also saves users the need to build the tools from source code, which is often the primary way the tools are distributed.

Finally, once a block or workflow has been built, it can be shared with users of the same e-SC instance, e.g. to build new workflows and pipelines. However, it can also be reused in other e-SC installations. Conversely, moving shell scripts between HPC systems would almost always involve some effort to rewrite and adapt them to particular configuration of a cluster (c.f. tool versions, paths to data, job submission parameters).

Nonetheless, the migration to workflows and the cloud involves also some disadvantages. Two of them seem to be most important.

First, the dataflow model we used imposes some restrictions in the way pipelines can be designed. As shown earlier, to escape these restrictions users can implement custom blocks. That, however, requires programming skills. Also being very specific to a particular use case, these blocks can hardly be reused in other contexts. On the other hand, the presented pipeline required us to implement only one such custom block, which does not justify the use of the more complex control-flow model. And to minimize the skillset required, e-SC supports a few scripting languages in which the blocks can be programmed (JavaScript, R, Octave). It also offers a web-based interface to develop simple blocks.

The other issue that will need our attention in the close future is the expected lower performance of cloud deployments when compared to traditional HPC clusters. This problem was observed earlier ([24], [25]) yet with the focus on the Amazon EC2 cloud and standard VM instances. To run our pipeline we use, however, the IaaS over Windows Azure for which performance comparison with HPC systems is not readily available. Moreover, recently both Microsoft and Amazon announced availability of HPC clusters in their clouds, a setting that closely resembles traditional HPC solutions. We are very keen to compare performance of the pipeline running in our HPC cluster and in the Azure cloud. Worth noting is also the fact that despite standard HPC systems are superior in raw performance, cloud clusters may produce better turnaround times [**?**]. This is because a traditional HPC cluster is shared among multiple users and almost always involves queue wait time. Conversely, a cluster started in the cloud can be dedicated for a single user only.

V. Conclusion

In this paper we describe our initial experiences with the migration of a WES pipeline from a script-based HPC environment to a workflow enactment system running in the cloud. Although it is too early to claim the absolute benefits of such a move, which need to be supported by collecting experiences over time, we expect that the workflow-based solution in the cloud will alleviate most of the issues we have been facing using the previous approach.

First of all, the new approach should allow us to track all the details of the variant discovery process. Having provenance data, we can explore how pipeline parameters influence results, compare and choose the best configurations but also give detailed evidence of each step in this process. The workflow-based approach will also ease the evolution of the pipeline, which is critical since new improved tools, tool versions and reference data appear regularly. It will also be important to facilitate pipeline redesign once the third generation sequencing becomes a viable option.

Following the move, our next goal for the future is twofold. First, we need to test performance of the cloud-based approach and compare it with the existing HPC-based solution. Despite our preliminary experiments show lower raw performance, which is in line with observations made by others previously, only the extensive testing and tuning of the new solution can reveal the exact difference. Yet with easy way to acquire large amount of resources and the new HPC facilities offered in the cloud we look at this exercise with hope.

Second, the existing workflow-based solution will become an input for the following phase involving variant filtration and interpretation. The basic requirements set in this work, such as flexibility and traceability, will play a very important role in that phase.

Acknowledgment

This work is sponsored by Biomedical Research Centre, National Institute for Health and Research and a grant from Windows Azure Research programme.

References

[1] L. D. Stein, "The case for cloud computing in genome informatics." *Genome biology*, vol. 11, no. 5, p. 207, Jan. 2010. [Online]. Available: http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=2898083&tool=pmcentrez

[2] S. B. Ng, E. H. Turner, P. D. Robertson, S. D. Flygare, A. W. Bigham, C. Lee, T. Shaffer, M. Wong, A. Bhattacharjee, E. E. Eichler, M. Bamshad, D. A. Nickerson, and J. Shendure, "Targeted capture and massively parallel sequencing of 12 human exomes." *Nature*, vol. 461, no. 7261, pp. 272–6, Sep. 2009. [Online]. Available: http://dx.doi.org/10.1038/nature08250

[3] S. Pabinger, A. Dander, M. Fischer, R. Snajder, M. Sperk, M. Efremova, B. Krabichler, M. R. Speicher, J. Zschocke, and Z. Trajanoski, "A survey of tools for variant analysis of next-generation genome sequencing data." *Briefings in bioinformatics*, pp. bbs086–, Jan. 2013. [Online]. Available: http://bib.oxfordjournals.org/content/early/2013/01/21/bib.bbs086.long

[4] I. B. Blanquer, G. Brasche, J. Caa, F. Gagliardi, D. Gannon, H. Hiden, H. Soncu, K. Takeda, A. Tomás, and S. Woodman, "Supporting NGS Pipelines in the cloud," *EMBnet.journal*, vol. 19, pp. 14–16, 2013. [Online]. Available: http://journal.embnet.org/index.php/embnetjournal/article/view/625/879

[5] S. Woodman, H. Hiden, P. Watson, and P. Missier, "Achieving Reproducibility by Combining Provenance with Service and Workflow Versioning," in *Procs. WORKS 2011*, Seattle, WA, USA, 2011.

[6] L. Moreau, B. Clifford, J. Freire, J. Futrelle, Y. Gil, P. Groth, N. Kwasnikowska, S. Miles, P. Missier, J. Myers, B. Plale, Y. Simmhan, E. Stephan, and J. Van Den Bussche, "The Open Provenance Model — Core Specification (v1.1)," *Future Generation Computer Systems*, vol. 7, no. 21, pp. 743–756, 2011.

[7] P. Missier, S. Woodman, H. Hiden, and P. Watson, "Provenance and data differencing for workflow reproducibility analysis," *Concurrency and Computation: Practice and Experience*, vol. in Press, pp. n/a—-n/a, 2013. [Online]. Available: http://dx.doi.org/10.1002/cpe.3035

[8] Y.-C. Lin, C.-S. Yu, and Y.-J. Lin, "Enabling large-scale biomedical analysis in the cloud." *BioMed research international*, vol. 2013, p. 185679, Jan. 2013. [Online]. Available: http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=3832998&tool=pmcentrez

[9] E. S. Torstenson, B. Li, and C. Li, "ASAP: an environment for automated preprocessing of sequencing data." *BMC research notes*, vol. 6, no. 1, p. 5, Jan. 2013. [Online]. Available: http://www.biomedcentral.com/1756-0500/6/5

[10] J. Goecks, A. Nekrutenko, and J. Taylor, "Galaxy: a comprehensive approach for supporting accessible, reproducible, and transparent computational research in the life sciences." *Genome biology*, vol. 11, no. 8, p. R86, Jan. 2010. [Online]. Available: http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=2945788&tool=pmcentrez&rendertype=abstract

[11] E. Afgan, D. Baker, N. Coraor, H. Goto, I. M. Paul, K. D. Makova, A. Nekrutenko, and J. Taylor, "Harnessing cloud computing with Galaxy Cloud." *Nature biotechnology*, vol. 29, no. 11, pp. 972–4, Nov. 2011. [Online]. Available: http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=3868438&tool=pmcentrez&rendertype=abstract

[12] E. Afgan, B. Chapman, and J. Taylor, "CloudMan as a platform for tool, data, and analysis distribution." *BMC bioinformatics*, vol. 13, no. 1, p. 315, Jan. 2012. [Online]. Available: http://www.biomedcentral.com/1471-2105/13/315

[13] P. Missier, S. Soiland-Reyes, S. Owen, W. Tan, A. Nenadic, I. Dunlop, A. Williams, T. Oinn, and C. Goble, "Taverna, reloaded," in *Procs. SSDBM 2010*, M. Gertz, T. Hey, and B. Ludaescher, Eds., Heidelberg, Germany, 2010. [Online]. Available: http://www.ssdbm2010.org/

[14] M. Abouelhoda, S. A. Issa, and M. Ghanem, "Tavaxy: integrating Taverna and Galaxy workflows with cloud computing support." *BMC bioinformatics*, vol. 13, p. 77, Jan. 2012. [Online]. Available: http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=3583125&tool=pmcentrez&rendertype=abstract

[15] R. K. Madduri, P. Dave, D. Sulakhe, L. Lacinski, B. Liu, and I. T. Foster, "Experiences in building a next-generation sequencing analysis service using galaxy, globus online and Amazon web service," *Proceedings of the Conference on Extreme Science and Engineering Discovery Environment Gateway to Discovery - XSEDE '13*, p. 1, 2013. [Online]. Available: http://dl.acm.org/citation.cfm?doid=2484762.2484827

[16] S. Lampa, M. Dahlö, P. I. Olason, J. Hagberg, and O. Spjuth, "Lessons learned from implementing a national infrastructure in Sweden for storage and analysis of next-generation sequencing data." *GigaScience*, vol. 2, no. 1, p. 9, Jan. 2013. [Online]. Available: http://www.gigasciencejournal.com/content/2/1/9

[17] M. El-Kalioby, M. Abouelhoda, J. Krüger, R. Giegerich, A. Sczyrba, D. P. Wall, and P. Tonellato, "Personalized cloud-based bioinformatics services for research and education: use cases and the elasticHPC package." *BMC bioinformatics*, vol. 13 Suppl 1, no. Suppl 17, p. S22, Jan. 2012. [Online]. Available: http://www.biomedcentral.com/1471-2105/13/S17/S22

[18] S. Schönherr, L. Forer, H. Weiß ensteiner, F. Kronenberg, G. Specht, and A. Kloss-Brandstätter, "Cloudgene: a graphical execution platform for MapReduce programs on private and public clouds." *BMC bioinformatics*, vol. 13, no. 1, p. 200, Jan. 2012. [Online]. Available: http://www.biomedcentral.com/1471-2105/13/200

[19] A. ODriscoll, J. Daugelaite, and R. D. Sleator, "Big data, Hadoop and cloud computing in genomics," *Journal of Biomedical Informatics*, vol. 46, no. 5, pp. 774–781, 2013. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1532046413001007

[20] A. McKenna, M. Hanna, E. Banks, A. Sivachenko, K. Cibulskis, A. Kernytsky, K. Garimella, D. Altshuler, S. Gabriel, M. Daly, and M. A. DePristo, "The Genome Analysis Toolkit: a MapReduce framework for analyzing next-generation DNA sequencing data." *Genome research*, vol. 20, no. 9, pp. 1297–303, Sep. 2010. [Online]. Available: http://www.scopus.com/inward/record.url?eid=2-s2.0-77956295988&partnerID=tZOtx3y1

[21] H. Hiden, S. Woodman, P. Watson, and J. Caa, "Developing cloud applications using the e-Science Central platform," *Philosophical transactions. Series A, Mathematical, physical, and engineering sciences*, vol. 371, no. 1983, Jan. 2013. [Online]. Available: http://www.ncbi.nlm.nih.gov/pubmed/23230161

[22] J. Cala, P. Watson, and S. Woodman, "Cloud Computing for Fast Prediction of Chemical Activity," in *Procs. 2nd International Workshop on Cloud Computing and Scientific Applications (CCSA)*, Ottawa, Canada, 2012.

[23] E. Deelman, D. Gannon, M. Shields, and I. Taylor, "Workflows and e-Science: An overview of workflow system features and capabilities," *Future Generation Computer Systems*, vol. 25, no. 5, pp. 528–540, May 2009. [Online]. Available: http://linkinghub.elsevier.com/retrieve/pii/S0167739X08000861

[24] S. Ostermann, A. Iosup, N. Yigitbasi, R. Prodan, T. Fahringer, and D. Epema, "A Performance Analysis of EC2 Cloud Computing Services for Scientific Computing," in *Cloud Computing*, 2010, pp. 115–131.

[25] E. Walker, "benchmarking Amazon EC2 for high-performance scientific computing," *;login:*, vol. 33, no. 5, pp. 18–23, 2008.