8th International Workshop on Science Gateways (IWSG 2016), 8-10 June 2016

From the Desktop to the Grid and Cloud: Conversion of KNIME Workflows to WS-PGRADE

Luis de la Garza
Center for Bioinformatics
Dept. of Computer Science
University of Tübingen, Germany
delagarza@informatik.uni-tuebingen.de

Fabian Aicheler
Center for Bioinformatics
Dept. of Computer Science
University of Tübingen, Germany

Oliver Kohlbacher
Center for Bioinformatics
Dept. of Computer Science
Faculty of Medicine
Quantitative Biology Center, University of Tübingen
Max Planck Institute for Developmental Biology
Germany

***Abstract*—Computational analyses for research usually consist of a complicated orchestration of data flows, software libraries, visualization, selection of adequate parameters, etc. Structuring these complex activities into a collaboration of simple, reproducible and well defined tasks brings down complexity and increases reproducibility. This is the basic notion of workflows.**

Several workflow engines exist that allow users to create and execute workflows. Each of these workflow engines might offer unique features. In some cases, certain features offered by platforms are royalty-based, hindering use in the scientific community.

We present our efforts to convert whole workflows created in the Konstanz Information Miner Analytics Platform workflow engine to the Grid and Cloud User Support Environment workflow engine. We see the former as a great workflow editor due to its considerable user base and user-friendly graphical interface. We deem the latter as a great backend engine that is able to interact with most major distributed computing interfaces. We introduce work that provides a platform-independent tool representation, thus assisting in the conversion of whole workflows. We also present the challenges inherent to workflow conversion across systems, as well as the ones posed by the conversion between the chosen workflow engines, along with our proposed solution to overcome these challenges.

The combined features of these two platforms (i.e., intuitive workflow design on a desktop computer and royalty-free execution of workflows on distributed high performance computing interfaces) greatly benefit researchers and minimize time spent in technical chores not directly related to their area of research.

***Keywords*—WS-PGRADE, KNIME, conversion, fine-grained interoperability, workflow**

I. INTRODUCTION

Today, computers are essential in various scientific fields. Example domains requiring high-performance computing include vaccine design, meteorology and astrophysics, or the multidisciplinary field of bioinformatics. Here, the declining costs of both data generation and storage in the last few years [1] pushed bioinformaticians into using high-performance computing (HPC) resources such as grids and clouds.

Simultaneously, the scope of research is getting more and more refined and complex. As such, upholding the scientific method increases in difficulty: Assuring reproducibility, that is, being able to reproduce previously observed results when keeping all variables constant, can often be an arduous task. Consequently, journals and news outlets repeatedly reported cases of published but irreproducible results [2], [3], [4].

To tackle complexity, researchers often break down big, complicated analyses into smaller units of work that are easier to manage. These so-called tasks then perform one specific function. Tasks process some input along with controlling parameters to produce a defined output. Input usually takes the form of files, whereas output could also be for example a set of visualizations. The combination of tasks is often referred to as a workflow. Task outputs can be passed on as inputs to other tasks, defining an order of execution for each step of the comprising workflow. Adoption of workflows not only increases reproducibility but also offers the following benefits:

- Storage of intermediate results (e.g., for troubleshooting, additional analysis, bottleneck identification)
- Simplified substitution of single tasks (e.g., for benchmarking, testing purposes)
- Parallel execution of workflow branches (i.e., *parameter sweep*)
- Reusability of components
- Independent, parallel development of specialized tasks

A. *Workflow Interoperability and Conversion*

Throughout this work we will use workflow terminology and representation consistent with previous work [14]. Figures 1 and 2 briefly summarize this.

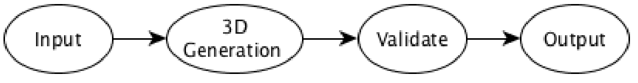


Fig. 1. **The *abstract* layer of a workflow.** Vertices represent tasks, edges indicate the execution order. At this point, no implementation or technical details are represented.

Since the *abstract* workflow layer contains solely application domain information, it is independent of the execution requirements. Thus, the *abstract* layer remains unchanged across workflow engines. In contrast, the *concrete* workflow layer, the workflow engine and the executing platform are tightly

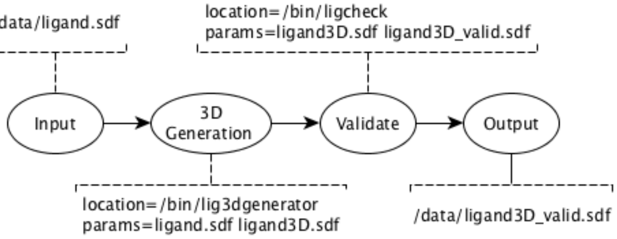

/data/ligand.sdf
location=/bin/ligcheck
params=ligand3D.sdf ligand3D_valid.sdf
Input
3D Generation
Validate
Output
location=/bin/lig3dgenerator
params=ligand.sdf ligand3D.sdf
/data/ligand3D_valid.sdf

Fig. 2. **The *concrete* layer of a workflow.** The *concrete* layer contains implicit application domain information. But unlike the *abstract* layer, vertices are annotated with extra attributes. These are the required resources to execute the portrayed tasks.

coupled. This divergence of *concrete* layers across engines makes workflow interoperability challenging. Furthermore, workflow engines often contain distinct features, complicating conversion across platforms.

One way to alleviate these problems is the development of platform-independent workflow representations, e.g., the Interoperable Workflow Intermediate Representation (IWIR) [10] and Yet another Workflow Language (YAWL) [11] to enable fine-grained interoperability (FGI). However, platform-independent workflow representations do not address workflow implementations. The Sharing interoperable Workflows for large-scale scientific Simulation on available distributed computing interfaces project (SHIWA) [13], for instance, provides execution of workflows built on different workflow engines by uploading them to the SHIWA Simulation Platform. Users handling data subject to privacy restrictions (e.g., patient data) might find it an unsuitable solution.

A proper workflow conversion from one workflow engine to another requires that the *abstract* layer remains unchanged. Only then, source and target workflow can be considered logically equivalent. Automated conversion is challenging and attention to detail is critical. Aspects to consider are the location of resources, and how different engines implement logical constructs (e.g., *parameter sweep*). Features offered in the source engine but not the target engine also represent a major complication. Figure 3 shows an example of a simplified workflow conversion.

II. IMPLEMENTATION

The Web Services Parallel Grid Runtime and Developer Environment Portal (WS-PGRADE) [8] is a web-based workflow engine that interacts with several major resource managers (e.g. UNICORE, LSF, Moab) to access distributed computing interfaces (DCIs). This makes it a great *back-end* workflow execution platform. Tasks of the same workflow can be executed on different DCIs. However, workflow creation is a multi-step process, posing problems for users without adequate training.

The Konstanz Information Miner Analytics Platform (KNIME Analytics Platform) [9], is hosted on the user's personal computer. The KNIME Analytics Platform features an intuitive interface, contains more than 1,000 tools and

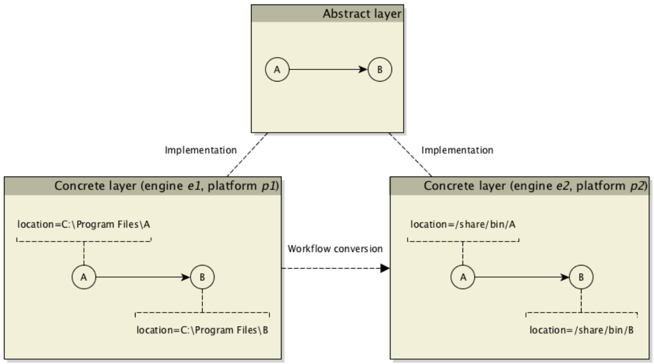

Abstract layer
A
B
Implementation
Implementation
Concrete layer (engine e1, platform p1)
location=C:\Program Files\A
A
B
location=C:\Program Files\B
Workflow conversion
Concrete layer (engine e2, platform p2)
location=/share/bin/A
A
B
location=/share/bin/B

Fig. 3. **Workflow conversion challenges.** Two different engines (i.e., *e1*, *e2*) running on two different platforms (i.e., *p1*, *p2*) contain different *concrete* layers of the same workflow. The *abstract* layer, however, remains unchanged. A successful workflow conversion must take into account not only the differences among the source and target engines, but must also consider the source and target platforms or operating systems.

hundreds of sample workflows. However, the addition of new tools requires knowledge of the Java programming language—an aspect that might keep some users away from this feature. A couple of royalty-based variants (i.e., the so-called *KNIME Collaborative Extensions*) are offered to remotely executed workflows, however, WS-PGRADE offers a wider support for resource managers to access DCIs.

We focus on providing fine-grained interoperability between a great workflow editor such as the KNIME Analytics Platform and a versatile, scalable workflow execution platform such as WS-PGRADE.

The first step to provide interoperability is to represent tasks in a platform-independent manner. Certain attributes of tool execution remain unchanged across platforms (e.g., version and parameters), while some others change (e.g., location of executables, input and output files). Attributes in need of adjustment have to be identified. A platform-independent tool representation facilitates the task conversion across platforms and thus the conversion of full workflows.

One of the first challenges in the conversion between these engines is the maintenance of a database that relates tools on the user's computer with tools on each of the target DCI platforms. The next set of challenges concerns the implementation of logical workflow constructs. The KNIME Analytics Platform implements *parameter sweep* via node-delimited workflow sections (i.e., using *ZipLoopStart*, *ZipLoopEnd* nodes). WG-PGRADE delimits such sections with *generator* and *collector* ports. Furthermore, WS-PGRADE allows users to assign data files to input ports. The KNIME Analytics Platform, however, requires a dedicated node (e.g., *Input File*, *Input Files*), whose output port transfers results to any connected input port.

Some features present in the KNIME Analytics Platform are not found in WS-PGRADE. The former requires ports to declare which data types they are compatible with (e.g., list of files); the latter is more flexible and lacks native support of file lists as inputs (i.e., each input or output port is related to one file). Different to WS-PGRADE, *KNIME Nodes* produce outputs not only via output ports: They can also set *flow variables*, which can be read further down the execution flow.

The KNIME Analytics Platform is a Java program with a graphic interface. *KNIME Nodes* are then instances of Java classes that live inside the Java process which launched the KNIME Analytics Platform. In other words, they require a running instance of the KNIME Analytics Platform to be executed, making their execution on a DCI a challenge.

The following sections describe our approach to address the mentioned challenges.

A. Disparities between KNIME and WS-PGRADE Workflows

KNIME workflows are composed of so-called *KNIME Nodes*. Each node defines a fixed number of input and output ports, each port corresponding to a *port type*. Port types are analogous to content types (e.g., *xls*, *csv*, *pdb*). KNIME enforces that only ports of compatible types can be interconnected. Furthermore, *KNIME Nodes* rely on the assumption that incoming and outgoing data are arranged in custom in-memory *data tables*. Each *KNIME Node* iterates over the rows of incoming data and is able to modify the contents of the input table, as well as its structure (e.g., by adding new columns or rows). Handling of arbitrary files in KNIME is done by passing around these same data tables, whose cells contain uniform resource identifiers (URI) pointing to the needed files.

WS-PGRADE, on the other hand, assigns ports directly to files and there is no type checking: any output port can be connected to any input port. Additionally, the structure of the incoming and outgoing files is arbitrary. This discrepancy makes fine-grained interoperability a challenge.

B. Extending the KNIME Analytics Platform

New tools can be added to the KNIME Analytics Platform, but this process requires knowledge of the Java programming language. Generic KNIME Nodes (GKN) [14] was developed to support tool import without programming experience. GKN allows arbitrary command line tools to behave as *KNIME Nodes* and to seamlessly interact with other nodes inside the KNIME Analytics Platform. The only requirement is the representation of the tools by Common Tool Descriptors (CTDs), which are XML files describing the inputs, outputs and parameters of a tool [14]. Currently, several software suites [15], [16], [17] are able to parse and generate CTDs (i.e., they are *CTD-enabled*). Figure 4 illustrates how CTDs interact with *CTD-enabled* tools.

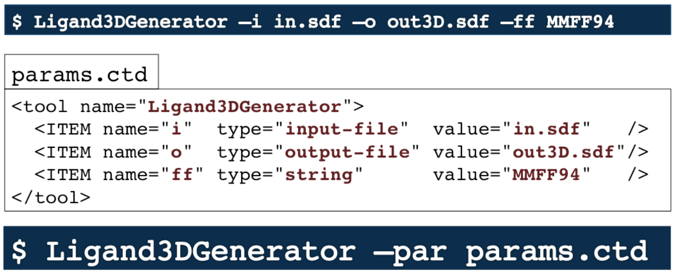
C. Conversion of Nodes

Conversion of *KNIME Nodes* that were imported using GKN is somewhat trivial. Each of these nodes represents an external tool that is independent of the KNIME Analytics Platform. In this case, the matching binary for the represented tool is required on each of the target DCIs.

We identify *native* nodes as those *KNIME Nodes* that were not imported using GKN. Each *native KNIME Node* is an instance of a Java class managed by the KNIME Analytics Platform. Such nodes exist only in the context of the JVM process that hosts the KNIME Analytics Platform. Execution of a single *KNIME Node* requires a running instance of the KNIME Analytics Platform and converting these nodes is not trivial. Furthermore, a suitable distribution of the KNIME Analytics Platform must be present on each of the target DCIs.

Data between *KNIME Nodes* can only be channeled between ports with compatible data types. Since channeled data are in-memory representations of table-formatted data as consumed and produced by *KNIME Nodes*, we have devised a solution that allows *native KNIME Nodes* to be executed *as if* they were command line tools: During the export process, *native KNIME Nodes* are individually packed into a small KNIME workflow. Each such generated workflow contains a copy of the original node, along with any user-established settings. Since inputs and outputs for the exported node won't be channeled *inside* an instance of the KNIME Analytics Platform, extra reader and writer nodes (i.e., *Table Reader* and *Table Writer*) are also included in this small workflow. This allows the serialization and deserialization of the in-memory data format required by *native KNIME Nodes*. The KNIME Analytics Platform can execute workflows in a so-called *batch mode*, without the need of a graphical user interface. A suitable command line is automatically generated during our export process.

Figures 5 and 6 depict how the conversion of these two types of nodes is performed.



```
$ Ligand3DGenerator –i in.sdf –o out3D.sdf –ff MMFF94
```

```
params.ctd
```

```
<tool name="Ligand3DGenerator">
  <ITEM name="i"  type="input-file"  value="in.sdf"   />
  <ITEM name="o"  type="output-file" value="out3D.sdf"/>
  <ITEM name="ff" type="string"      value="MMFF94"   />
</tool>
```

```
$ Ligand3DGenerator –par params.ctd
```

Fig. 4. **A CTD in action.** Top section: parameters needed for the tool *Ligand3DGenerator* [17] to be executed, namely input file, output file and the desired force field. Middle section: a sample CTD snippet representing an execution of the *Ligand3DGenerator* using the shown parameter values. Bottom section: a *CTD-enabled* tool with the given sample CTD.

D. Conversion of Workflows: Exporting KNIME wofkflows to WS-PGRADE

The KNIME2gUSE plug-in [14] is fully integrated within the KNIME Analytics Platform and allows users to export workflows by using an intuitive user interface. The export results in a file that can be imported into WS-PGRADE, ready to be executed on any DCI configured in WS-PGRADE with minor modifications.

Instead of developing a series of converters whose output can be executed by resource or cloud managers, we chose WS-PGRADE as the target engine for the export process. This was done because WS-PGRADE directly interacts with

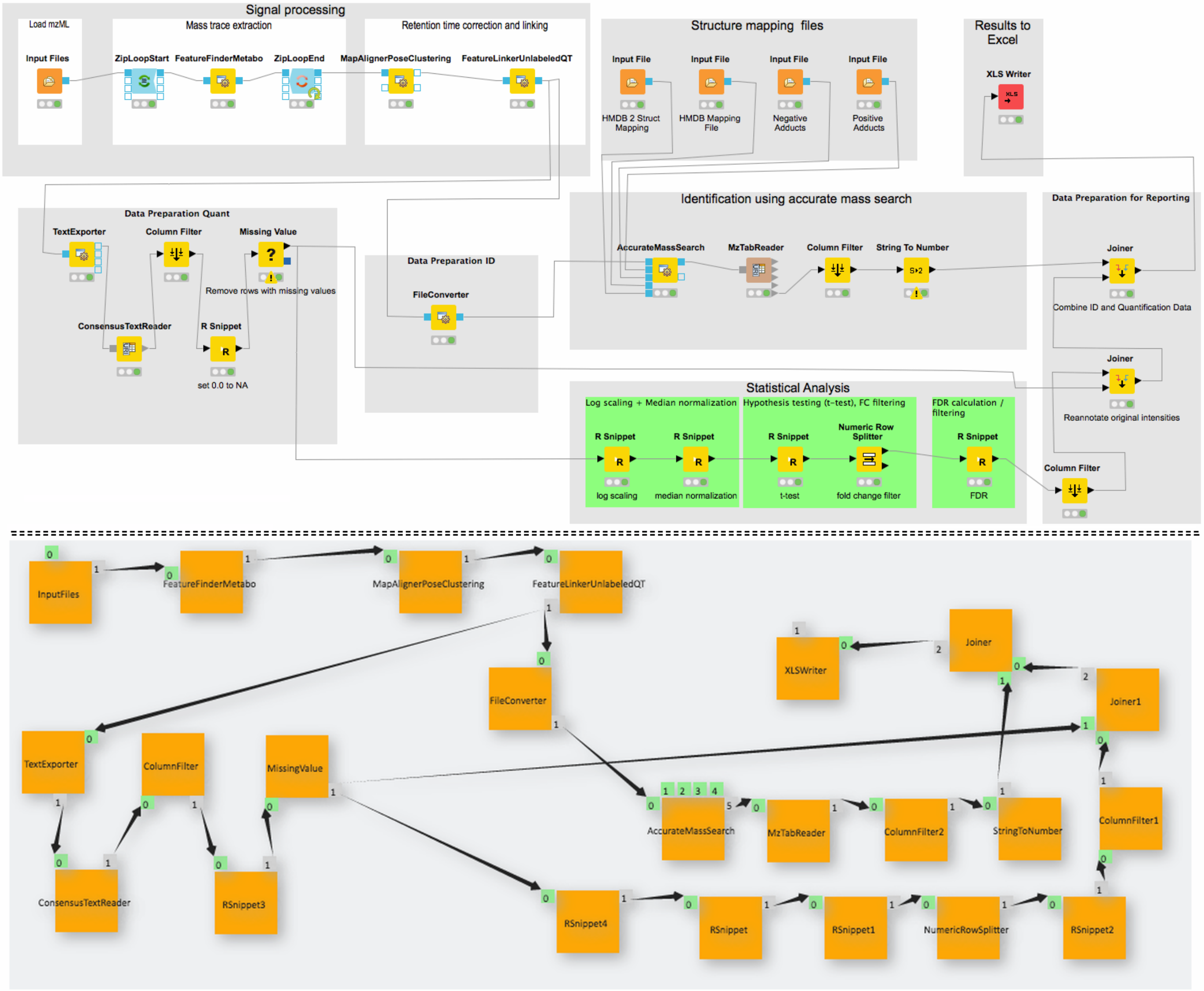


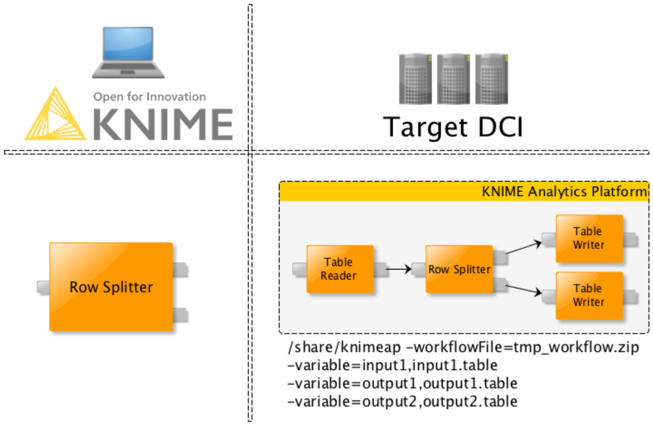
Fig. 7. **Metabolics biomarker discovery workflow both in the KNIME Analytics Platform and WS-PGRADE**. Top: workflow implemented in the KNIME Analytics Platform, OpenMS [15] was imported using GKN. Bottom: workflow generated by KNIME2gUSE imported into WS-PGRADE (workflow slightly edited for visual clarity). Even if some elements are missing after conversion (e.g., *ZipLoopStart*, *ZipLoopEnd*, *Input File* nodes), both versions have the same *abstract* layer. This is due to the difference in implementation of logically equivalent constructs in WS-PGRADE and the KNIME Analytics Platform, such as *parameter sweep*.

several resource and cloud managers. It also features workflow submission, control, monitoring and statistics. These are functionalities which resource managers or cloud engines often lack.

Some features present in the KNIME Analytics Platform do not exist in WS-PGRADE (e.g., *KNIME Flow Variables*). As such, these must be properly emulated. For instance, the KNIME Analytics Platform natively supports the association of single input/output ports to a file list determined at *runtime*, a feature not present in WS-PGRADE. To overcome this, a wrapper script is automatically generated by KNIME2gUSE that zips corresponding files into a single archive. To translate *parameter sweep* sections, conversion removes KNIME Analytics Platform *ZipLoopStart* and *ZipLoopEnd* nodes and substitutes suitable WS-PGRADE *generator* and *collector* ports.

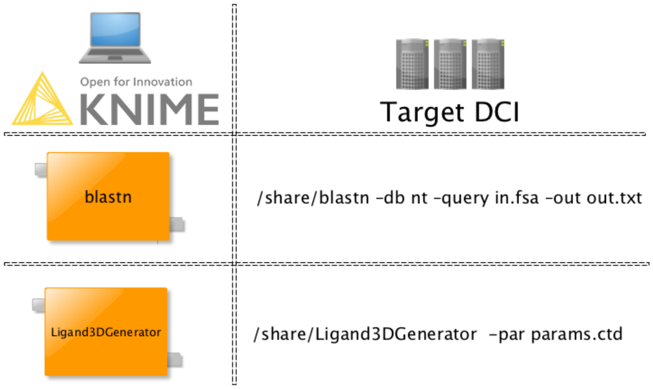
E. Example Application: Biomarker Discovery in Metabolomics

Metabolomics is a mass spectrometry-based approach aimed to evaluate the entirety of a metabolite sample. Applications include the tracking of chemicals and their transformation products in waste water [18], identification of cancer types via biomarkers [19], [20] and elucidation of disease-underlying mechanisms [21]. Compared to complementary *omics* technologies (e.g., transcriptomics, proteomics), metabolomics is



```
/share/knimeap -workflowFile=tmp_workflow.zip
-variable=input1,input1.table
-variable=output1,output1.table
-variable=output2,output2.table
```

Fig. 5. **Conversion of *native* KNIME nodes.** Since *Row Splitter* exists only in the context of the KNIME Analytics Platform, it requires an instance of a KNIME Analytics Platform for its execution. KNIME2gUSE generates a workflow containing the required input/output nodes and a copy of *Row Splitter* with the same configuration settings as its origin node. The generated command line invokes a KNIME Analytics Platform in the target DCI in *batch mode*.



```
/share/blastn -db nt -query in.fsa -out out.txt
/share/Ligand3DGenerator -par params.ctd
```

Fig. 6. **Conversion of GKN-imported nodes.** The nodes depicted on the left side directly interact with binaries located on the user's desktop computer. The right column shows the mapped binaries and an equivalent execution on a target DCI. Since *Ligand3DGenerator* is *CTD-enabled*, a suitable CTD file can be generated; this is not the case for the *blastn* tool.

closer to the actual biochemical processes that occur, making it attractive for biomarker development.

A common analysis approach for studies interested in comparative metabolite concentrations is label-free quantification. The independence from chemical labels allows the direct comparison of small molecules across an arbitrary number of samples. As a consequence, the need to evaluate hundreds of gigabyte-sized samples in concert is already common. Numbers and sizes of concurrently evaluated samples are steadily increasing, emphasizing the necessity for distributed computing.

We provide an example workflow for metabolomics biomarker discovery using OpenMS [15] for mass spectrometry algorithms as well as various *native KNIME Nodes* (including nodes for the R scripting language). The KNIME workflow and its converted WS-PGRADE version are shown in Figure 7. We assume some initial preparations were performed prior to the execution of the workflow, namely, conversion from closed mass spectrometer vendor formats to the open *mzML* format and data reduction by means of peak picking, which could also be implemented in KNIME via OpenMS [15] tools.

Using a detection method for so-called small molecules [22], we adapted a label-free quantification pipeline [23]. The quantification part of our biomarker discovery workflow consists of sample specific feature detection (i.e., finding the convex hulls and respective centroids of analyte mass traces) followed by temporal alignment of samples and the quantification of corresponding features across samples.

Downstream small molecule identification was done via mass-based search in the Human Metabolome Database. Included sample normalization allows for comparison of analyte abundances across samples. Analytes whose abundances vary significantly after false discovery rate correction are annotated with the mass-based identifications and exported to a Microsoft Excel Spreadsheet (XLS format).

III. FUTURE WORK

The KNIME Analytics Platform features *Metanodes* encapsulating complete workflows. We would like to extend KNIME2gUSE to support their conversion. Furthermore, seeing that considerable effort has been put into creating platform-independent workflow representation formats, we would like to add IWIR and YAWL file generation to KNIME2gUSE. We would also like to implement a similar converter, KNIME2Galaxy.

IV. CONCLUSION

Workflows assist reproducibility and minimize time spent validating research by reducing analysis complexity. There are currently several workflow engines with user-friendly interfaces that support remote execution of workflows. However, we feel that their scalability and support of major resource managers is still lacking. In contrast, HPC infrastructures and their resource managers rarely support the execution and control of workflows. As a consequence, HPC users often require programming skills to handle the channeling of data as well as to submit, monitor and control the respective computing jobs.

Here we present our efforts to support workflow export from the KNIME Analytics Platform to WS-PGRADE. We identified challenges for the corresponding whole workflow conversion and detailed our solutions. Our export plugin brings a user-friendly, easy-to-install and intuitive workflow engine for personal computers together with an easy-to-install and scalable HPC workflow platform that supports several resource managers.

We thus provide the individual advantages of both engines without any of their shortcomings. Overall, our methods decrease time spent designing workflows and troubleshooting conversion for different workflow engines.

ACKNOWLEDGMENT

The authors would like to thank Bernd Wiswedel, Thorsten Meinl, Patrick Winter and Michael Berthold for their support, patience and help in developing the KNIME2gUSE extension.

This work was supported by the German Network for Bioinformatics Infrastructure (Deutsches Netzwerk für Bioinformatik-Infrastruktur, de.NBI).

REFERENCES

[1] C. S. Greene, J. Tan, M. Ung, J. H. Moore, and C. Cheng, "Big data bioinformatics." *Journal of cellular physiology*, vol. 229, no. 12, pp. 1896–900, Dec. 2014.

[2] M. McNutt, "Reproducibility." *Science (New York, N.Y.)*, vol. 343, no. 6168, p. 229, 2014.

[3] "Trouble at the lab," *The Economist*, oct 2013.

[4] M. Baker, "Over half of psychology studies fail reproducibility test," *Nature*, Aug. 2015.

[5] W. M. P. Van der Aalst, "The Application of Petri Nets to Workflow Management," pp. 21–66, 1998.

[6] D. Blankenberg, G. V. Kuster, N. Coraor, G. Ananda, R. Lazarus, M. Mangan, A. Nekrutenko, and J. Taylor, "Galaxy: A web-based genome analysis tool for experimentalists," 2010.

[7] P. Missier, S. Soiland-Reyes, S. Owen, W. Tan, A. Nenadic, I. Dunlop, A. Williams, T. Oinn, and C. Goble, "Taverna, reloaded," in *Lecture Notes in Computer Science*, vol. 6187 LNCS, 2010, pp. 471–481.

[8] P. Kacsuk, Z. Farkas, M. Kozlovszky, G. Hermann, A. Balasko, K. Karoczkai, and I. Marton, "WS-PGRADE/gUSE Generic DCI Gateway Framework for a Large Variety of User Communities," *Journal of Grid Computing*, vol. 10, no. 4, pp. 601–630, 2012.

[9] M. R. Berthold, N. Cebron, F. Dill, T. R. Gabriel, T. Kötter, T. Meinl, P. Ohl, K. Thiel, and B. Wiswedel, "KNIME - the Konstanz information miner," *ACM SIGKDD Explorations Newsletter*, vol. 11, no. 1, p. 26, Nov. 2009.

[10] K. Plankensteiner, J. Montagnat, and R. Prodan, "IWIR: A Language Enabling Portability Across Grid Workflow Systems," in *SIGMOD Rec.*, vol. 34, no. 3, 2011, pp. 97–106.

[11] W. van der Aalst and A. ter Hofstede, "YAWL: yet another workflow language," *Information Systems*, vol. 30, no. 4, pp. 245–275, Jun. 2005.

[12] M. Abouelhoda, S. Issa, and M. Ghanem, "Tavaxy: Integrating Taverna and Galaxy workflows with cloud computing support," p. 77, 2012.

[13] G. Terstyanszky, T. Kukla, T. Kiss, P. Kacsuk, A. Balasko, and Z. Farkas, "Enabling scientific workflow sharing through coarse-grained interoperability," *Future Generation Computer Systems*, vol. 37, pp. 46–59, 2014.

[14] L. de la Garza, J. Veit, A. Szolek, M. Röttig, S. Aiche, S. Gesing, K. Reinert, and O. Kohlbacher, "From the desktop to the grid: scalable bioinformatics via workflow conversion," *BMC Bioinformatics*, vol. 17, no. 1, pp. 1–12, 2016.

[15] M. Sturm, A. Bertsch, C. Gröpl, A. Hildebrandt, R. Hussong, E. Lange, N. Pfeifer, O. Schulz-Trieglaff, A. Zerck, K. Reinert, and O. Kohlbacher, "OpenMS - an open-source software framework for mass spectrometry." *BMC bioinformatics*, vol. 9, p. 163, 2008.

[16] A. Döring, D. Weese, T. Rausch, and K. Reinert, "SeqAn an efficient, generic C++ library for sequence analysis." *BMC bioinformatics*, vol. 9, no. 1, p. 11, Jan. 2008.

[17] A. Hildebrandt, A. K. Dehof, A. Rurainski, A. Bertsch, M. Schumann, N. C. Toussaint, A. Moll, D. Stöckel, S. Nickels, S. C. Mueller, H.-P. Lenhof, and O. Kohlbacher, "BALL–biochemical algorithms library 1.3." *BMC bioinformatics*, vol. 11, p. 531, 2010.

[18] E. L. Schymanski, H. P. Singer, P. Longrée, M. Loos, M. Ruff, M. A. Stravs, C. Ripollés Vidal, and J. Hollender, "Strategies to characterize polar organic contamination in wastewater: exploring the capability of high resolution mass spectrometry." *Environmental science & technology*, vol. 48, no. 3, pp. 1811–8, Jan. 2014.

[19] M. Sugimoto, D. T. Wong, A. Hirayama, T. Soga, and M. Tomita, "Capillary electrophoresis mass spectrometry-based saliva metabolomics identified oral, breast and pancreatic cancer-specific profiles." *Metabolomics : Official journal of the Metabolomic Society*, vol. 6, no. 1, pp. 78–95, mar 2010.

[20] C. Denkert, J. Budczies, T. Kind, W. Weichert, P. Tablack, J. Sehouli, S. Niesporek, D. Könsgen, M. Dietel, and O. Fiehn, "Mass spectrometry-based metabolic profiling reveals different metabolite patterns in invasive ovarian carcinomas and ovarian borderline tumors." *Cancer research*, vol. 66, no. 22, pp. 10 795–804, Nov. 2006.

[21] J. S. Hansen, X. Zhao, M. Irmler, X. Liu, M. Hoene, M. Scheler, Y. Li, J. Beckers, M. Hrab? de Angelis, H.-U. Häring, B. K. Pedersen, R. Lehmann, G. Xu, P. Plomgaard, and C. Weigert, "Type 2 diabetes alters metabolic and transcriptional signatures of glucose and amino acid metabolism during exercise and recovery." *Diabetologia*, vol. 58, no. 8, pp. 1845–54, Aug. 2015.

[22] E. Kenar, H. Franken, S. Forcisi, K. Wörmann, H.-U. Häring, R. Lehmann, P. Schmitt-Kopplin, A. Zell, and O. Kohlbacher, "Automated label-free quantification of metabolites from liquid chromatography-mass spectrometry data." *Molecular & cellular proteomics : MCP*, vol. 13, no. 1, pp. 348–59, jan 2014.

[23] H. Weisser, S. Nahnsen, J. Grossmann, L. Nilse, A. Quandt, H. Brauer, M. Sturm, E. Kenar, O. Kohlbacher, R. Aebersold, and L. Malmström, "An automated pipeline for high-throughput label-free quantitative proteomics." *Journal of proteome research*, vol. 12, no. 4, pp. 1628–44, Apr. 2013.