Journal of proteome research

Article

pubs.acs.org/jpr

Pladipus Enables Universal Distributed Computing in Proteomics Bioinformatics

Kenneth Verheggen,[†,‡,§] Davy Maddelein,[†,‡,§] Niels Hulstaert,[†,‡,§] Lennart Martens,[*,†,‡,§] Harald Barsnes,[∥,⊥] and Marc Vaudel[∥]

[†]Medical Biotechnology Center, VIB, Albert Baertsoenkaai 3, Ghent B-9000, Belgium

[‡]Department of Biochemistry, Ghent University, Albert Baertsoenkaai 3, Ghent B-9000, Belgium

[§]Bioinformatics Institute Ghent, Ghent University, Albert Baertsoenkaai 3, Ghent B-9000, Belgium

[∥]Proteomics Unit, Department of Biomedicine, University of Bergen, Postboks 7804, N-5020 Bergen, Norway

[⊥]KG Jebsen Center for Diabetes Research, Department of Clinical Science, University of Bergen, Postboks 7804, N-5020 Bergen, Norway

ABSTRACT: The use of proteomics bioinformatics substantially contributes to an improved understanding of proteomes, but this novel and in-depth knowledge comes at the cost of increased computational complexity. Parallelization across multiple computers, a strategy termed distributed computing, can be used to handle this increased complexity; however, setting up and maintaining a distributed computing infrastructure requires resources and skills that are not readily available to most research groups. Here we propose a free and open-source framework named Pladipus that greatly facilitates the establishment of distributed computing networks for proteomics bioinformatics tools. Pladipus is straightforward to install and operate thanks to its user-friendly graphical interface, allowing complex bioinformatics tasks to be run easily on a network instead of a single computer. As a result, any researcher can benefit from the increased computational efficiency provided by distributed computing, hence empowering them to tackle more complex bioinformatics challenges. Notably, it enables any research group to perform large-scale reprocessing of publicly available proteomics data, thus supporting the scientific community in mining these data for novel discoveries.

KEYWORDS: *distributed computing, large scale data processing, cluster computing, parallelization*

INTRODUCTION

Various high-throughput methods for the analysis of proteomes are applied in modern day bioinformatics proteomics. For instance, in mass spectrometry (MS)-based proteomics, technical advances and novel methodologies lead to ever more samples and tissues being analyzed.[1] This increased complexity is illustrated by the growing project sizes in PRIDE.[2] Between 2010 and 2014, the median number of spectra for a project increased over 15-fold (from 16 787 to 256 801 spectra per project). The culmination of this expansion is found in proteome-wide studies such as the recent efforts to elaborate a comprehensive MS-based human proteome characterization.[3,4]

These studies are indicative of how data sets are becoming increasingly large and complex, thus requiring high-performance computational setups to be analyzed in a reasonable time. The required computational power can be attained through parallelization and scalable distributed computing similarly to other data intensive fields.[5] The former occurs by delegating calculations across multiple central processing units (CPUs) within the same computer; however, the number of CPUs included in a single computer is limited. The latter strategy on the contrary, which distributes tasks across a collection of networked computers, is often dubbed grid computing. This approach bypasses the limitations of the single computer and allows the wall time required to process data to be reduced; however, these solutions are often expensive and difficult to set up and maintain. Moreover, the continuous operation of a distributed computing network can be cumbersome and requires expert support.

Previous attempts at providing a user-friendly framework for grid based computing have proven successful. For example, the Galaxy framework[6] has been widely used for large-scale data processing in both genomics and proteomics.[7−11] The wide applicability of frameworks such as Galaxy is largely due to the possibility to add and develop custom tools; however, this generally requires expert support and the setup of an operational network requires either expensive external compute

Special Issue: Large-Scale Computational Mass Spectrometry and Multi-Omics

Received: September 14, 2015
Published: October 29, 2015

DOI: 10.1021/acs.jproteome.5b00850
J. Proteome Res. 2016, 15, 707−712

cloud providers or high-end hardware. Another successful platform for distributed computing is found in "Rosetta@home".[12] The project excels in using voluntarily donated compute CPU cycles to predict protein−protein docking. The follow-up project "Foldit" is a gamified version developed to achieve the same goals as its predecessor using crowdsourcing.[13] In theory, Rosetta can scale limitlessly and is purely dependent on how many computers are connected to the processing network. Despite having access to this enormous pool of resources, these applications are tailored to a specific niche (protein docking in this particular case) and substantial expertise is needed to adapt these approaches for novel applications.

Here we present an easily adoptable and generic platform for distributed software, called Pladipus. It provides an end-user-oriented solution to distribute bioinformatics tasks over a network of computers, managed through an intuitive graphical user interface (GUI). Pladipus aims to combine the versatility of available tools in Galaxy with the ability to tap into the processing power of multiple (idle) computers on a network. Research groups thus benefit from the power of distributed computing to run established, third-party as well as custom in-house algorithms on an inexpensive cluster infrastructure without additional cost or expert maintenance requirement.

EXPERIMENTAL PROCEDURES

A Pladipus network allows the user to execute workflows, here called *Runs*, in a distributed fashion. *Runs are* defined by a collection of *Jobs*, which in turn consist of a collection of *Processing Steps* to be executed by the distributed system, as illustrated in Figure 1. Machines in such a network come as two types: (i) *Managers* that manage the execution of *Runs* and display progress to the user and (ii) *Executors* that perform the actual execution of requested *Runs*. Notably, the network can consist of multiple *Managers* and *Executors*, and a single machine can be both *Manager* and *Executor*. The central hub of a Pladipus network is a combination of ActiveMQ and MySQL. The former serves as the message bus, over which tasks and their statuses are communicated to the Pladipus network. The latter is used as the persistent memory for the Pladipus *Runs* and their configuration. This infrastructure allows for multiple management instances to connect to the Pladipus network, simultaneously providing multiple versatile and user-friendly access points to the network.

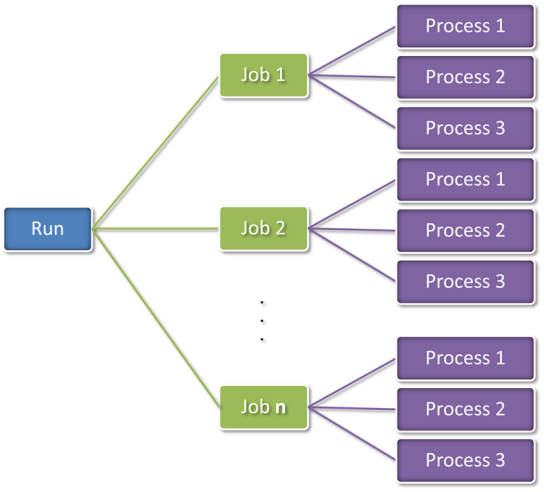

Figure 1. Standard Pladipus *Run* consists of several *Jobs*, each of which consists of a sequential collection of customizable *Processing Steps*.

Example data sets are provided on the Pladipus Web site, while example processing modules are included in the Pladipus installation. For the database searches, the example data sets consist of a subset of the human proteome draft by Kim et al.[4] as peak lists in the Mascot generic file (mgf) format obtained using msconvert as part of ProteoWizard[14] version 3.0.8789, along with the reported search parameters and a concatenated target/decoy protein sequence database consisting of the human complement of the UniProtKB/SwissProt database[15] (version of August 2015) with common contaminants and reverse decoy sequences appended. The embedded modules work out of the box and include convenient access to numerous database search algorithms via SearchGUI[16] and PeptideShaker,[17] de novo sequencing using multiple algorithms via DeNovoGUI,[18] and three common forms of the BLAST[19] algorithm (blastn, blastp, and blastx).

Note that Pladipus is not limited to the embedded selection of tools and that other software and workflows can be added, including the addition of user developed custom algorithms. The inclusion of external tools or custom operations is done by writing a Java wrapper as detailed in the documentation (http://compomics.github.io/pladipus/wiki/4advancedoperations.html).

Pladipus is programmed in Java and is freely available as open source under the permissive Apache2 license. Source code, documentation including example files, installer, and a video tutorial can be found at https://compomics.github.io/projects/pladipus.html.

RESULTS AND DISCUSSION

The Pladipus platform and its documentation were designed to make the execution and management of distributed tasks available to all by greatly simplifying the computational skill requirements. As such, it forms a step in the direction of providing distributed computing to all research teams. The prerequisites for the computer network are listed on the wiki (http://compomics.github.io/pladipus/wiki/1installation.html#prerequisites). The skills required for managing and operating a Pladipus network scale with the desired level of operability: installing and running the embedded tools on a cluster of (desktop) computers only requires the installation of a basic MySQL server; operating from headless servers requires basic command line skills and knowledge of the operating system and implementation and support for custom tools benefits from basic programming skills.

Pladipus can easily be installed by downloading and running the latest version of the installer available at the tool's Web site (https://compomics.github.io/projects/pladipus.html) or by downloading and extracting a prepackaged zip file for headless installations. Note that the installation procedure includes installation of the third-party open source software packages ActiveMQ (http://activemq.apache.org) and MySQL (https://www.mysql.com). Once installed on the computers in a network and with the ActiveMQ and MySQL database running in the same network, Pladipus can be run in either *Management* mode for the user or *Execution* mode for the worker nodes that will execute the tasks. It is important to note that *Executer* nodes can be dynamically added to or removed from the Pladipus network (e.g., automatic start-up of the *Execution* mode on desktop computers in the lab at night and automatic

DOI: 10.1021/acs.jproteome.5b00850
J. Proteome Res. 2016, 15, 707−712

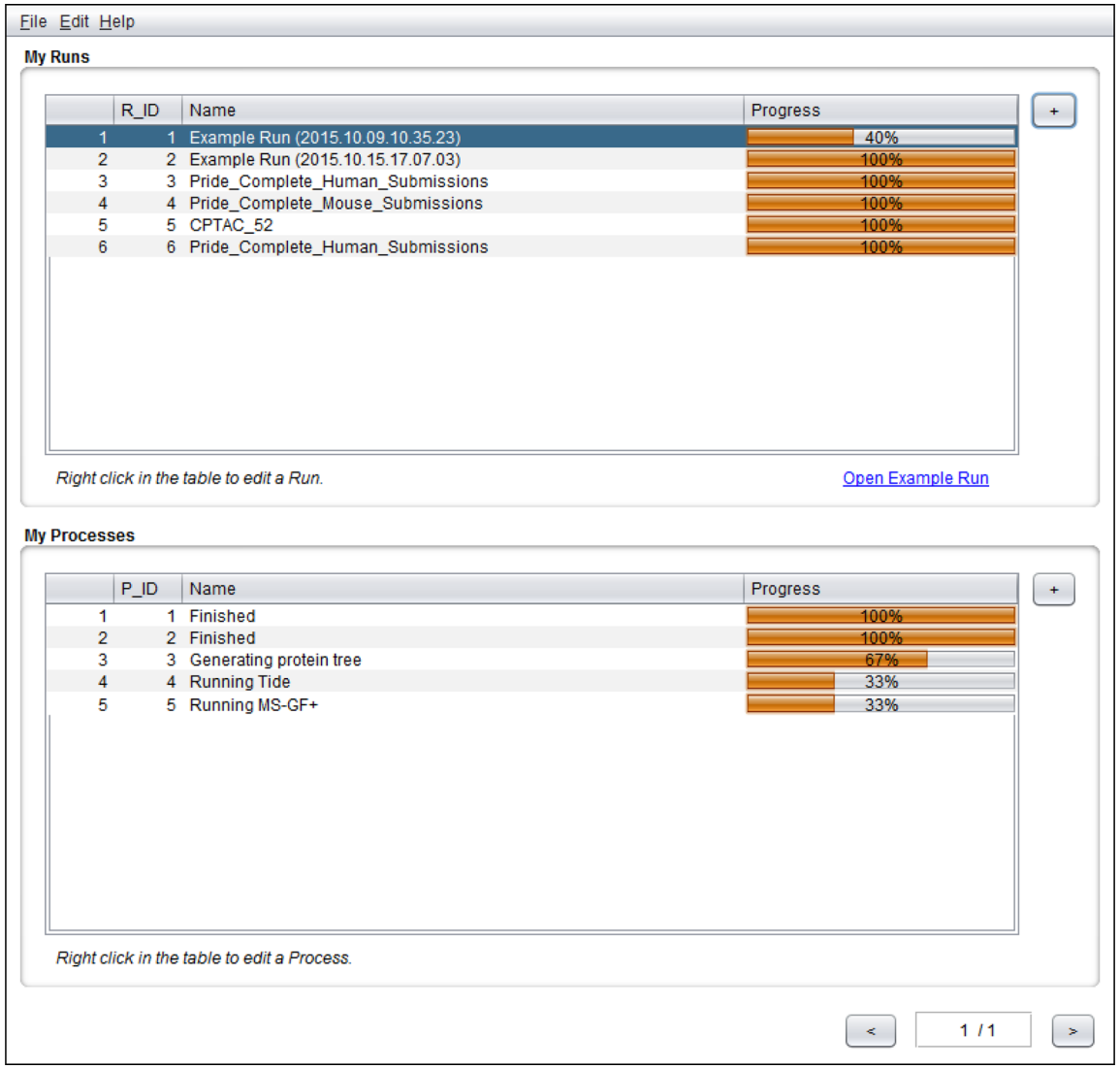

Figure 2. Main Pladipus interface allows the user to manage *Runs* and *Jobs*. Users can *Create*, *Launch*, *Reset*, or *Delete* these *Runs*. Additionally, individual *Processes* can be updated with new parameters.

shut-down in the morning). The *Management* mode provides a GUI that allows the user to set up and manage processing *Runs*, which can then be sent via ActiveMQ for execution across all nodes on the network that are running the headless *Executer* mode.

Pladipus can be deployed on both Unix and Windows operating systems and can seamlessly combine these in heterogeneous networks; however, the ability to operate across platforms may be a limitation of the specific tool that will be run on the Pladipus infrastructure. For example, vendor libraries for raw file parsing are only available on Windows operating systems, which, in turn, requires one or more Windows instances in the Pladipus network for the specific task of raw file conversion using Proteowizard.[14] The user can declare operating system restrictions at the job level in the *Job Requirements*. *Job Requirements* can also be used to specify architecture, storage, memory, and processor usage requirements. Prior to accepting a *Job*, a Pladipus *Worker* will first perform a system check against these *Job Requirements* to ensure it is capable of executing this job. In case the *Worker* does not comply with the *Job Requirements*, the *Job* will be handled by another *Worker*. For more details, please refer to the project's documentation page (http://compomics.github.io/pladipus/wiki/2manual.html).

To launch a new *Run* on the network, the user starts Pladipus in the *Management* mode and logs into the system with their credentials. The main display is then shown, where *Runs* can be created, managed, and launched (Figure 2). The intuitive GUI makes it simple to set up a new *Run*. It is also possible to create a *Run* by importing a template from a file, using one of several preset modules, or to create an entirely new template from scratch (Figure 3). Adding *Jobs* to the *Run* is done via the GUI by specifying the required input and parameters or by submitting these as a file. A *Job Configuration File* is automatically generated when a *Run* is created and stored.

Once the *Run* is submitted to the Pladipus controller, it will become visible in the *Run Manager* section, where the status of the individual *Jobs* can be monitored. This is important and useful because Pladipus is an asynchronous system, where submitted *Jobs* will be picked up as soon as a worker becomes available. The user can select a *Run* or a selection of *Jobs* and then right-click the selection to reveal a popup menu with the option to *Start*, *Cancel*, or *Delete* a *Run* or *Job*. When *Runs* are submitted to the Pladipus network, any connected, idle worker

DOI: 10.1021/acs.jproteome.5b00850
J. Proteome Res. 2016, 15, 707−712

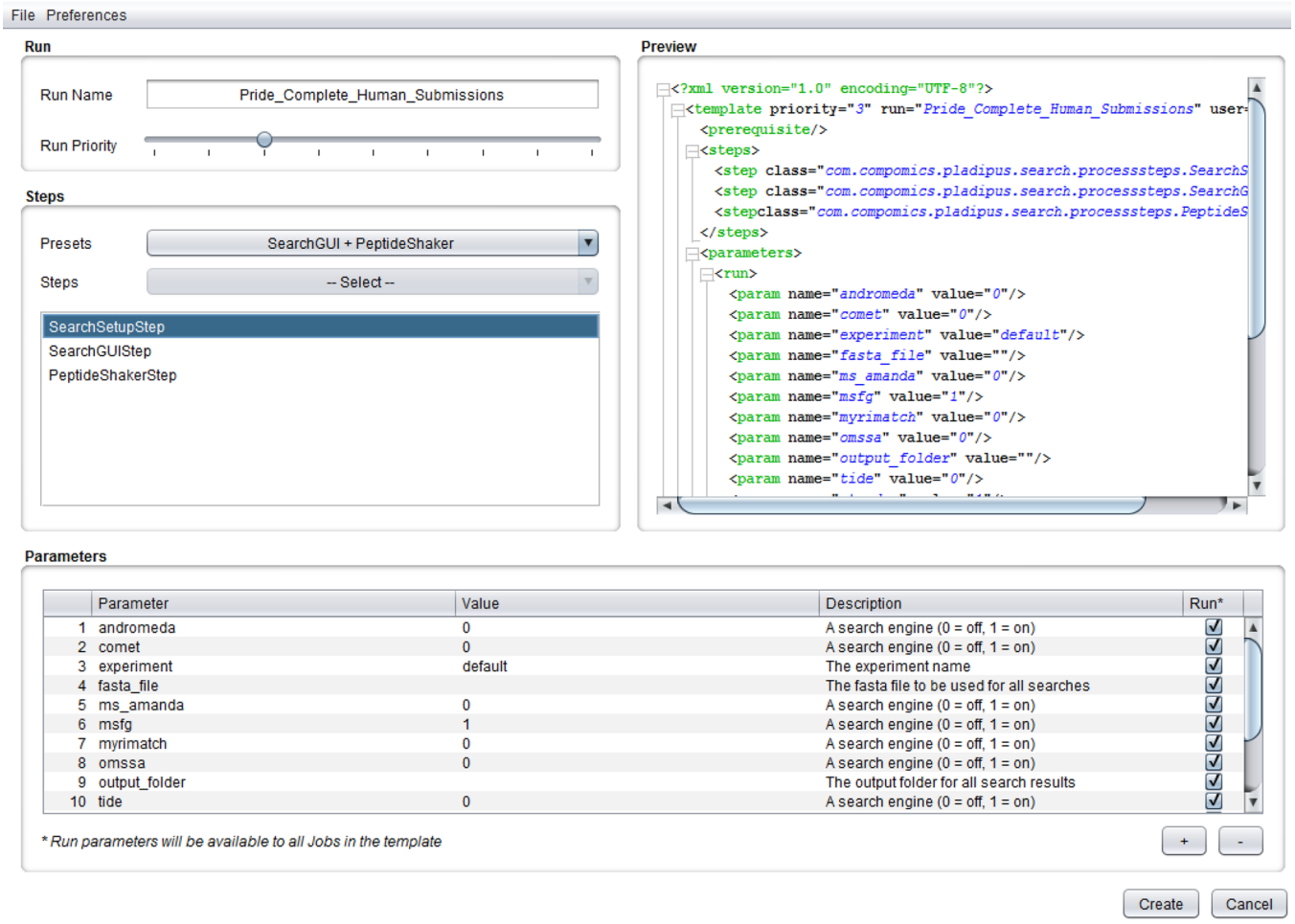

Figure 3. *Run Creation Wizard* allows users to create and customize template *Runs*. The *Run* section can be used to set a custom title. The *Steps* section shows the steps to be executed. In this example, the search is initialized in the *SearchSetupStep*, the searches are performed in the *SearchGUIStep*, and the results are postprocessed in the *PeptideShakerStep*. The *Parameters* section shows the required parameters to create processes. The *Preview* section shows a representation of the XML file that will be created along with the parameter configuration file.

instances will automatically retrieve these pending jobs and execute them. It should be noted that the MySQL database storage of *Runs* and *Jobs* allows the system to maintain a memory of all runs and jobs and their most recent state, even in the event of a system failure. When rebooted, the Pladipus system automatically resumes activity from where it left at shutdown or failure. To numerically assess the benefits of using a distributed computing framework, we searched 52 CPTAC experiments (LTQ-Study6: Orbitrap@86)[20] against a protein sequence database using various network pool sizes (1, 2, 5, 10, 20, and 40 computing instances) in triplicate. All Pladipus execution nodes were set up as identical copies (16 GB ram, 12 cores, 250 GB disk space, Ubuntu precise -12.04.5 LTS Linux operating system). The CPTAC raw files were converted to the Mascot generic format (MGF) using msconvert as part of the Proteowizard[14] package version 3.0.8789. The pipeline consisted of SearchGUI[16] version 2.0.4 for the search engines, followed by PeptideShaker[17] version 1.0.1 for integration and postprocessing of the search results. The sequence database used was the human complement of UniProtKB/SwissProt (release-2015_05), appended with the reversed protein sequences as decoys. A selection of three search engines was applied: X!Tandem,[21] Tide,[22] and MS-GF+.[23] The time required to process the entire CPTAC Study 6 data set was on average 813 ± 28 min on a single compute instance. When scaling the pool up to 40 identical computers, the wall time to process these data decreased to an average of 31 ± 1 min. As expected for a distributed system,[5] the wall time decreased nearly exponentially with the number of workers (Figure 4).

CONCLUSIONS

Pladipus empowers scientists to intuitively establish advanced workflows, allowing the execution of computationally intensive tasks via distributed computing on existing hardware without additional costs or specific expertise. These customized workflows can in turn be shared among laboratories and made available upon publication. This important feature promotes the reproducibility of results and can help foster across-field collaborations.

A good example use case is presented by the ability to collectively (re)process publicly available data as part of a community effort to uncover novel knowledge.[24] By taking advantage of the distribution of such intensive tasks across a dynamically scalable network of computers, Pladipus substantially reduces the wall time required to perform a broad range of proteomics bioinformatics tasks. This is illustrated by the near exponential decrease in the wall time when performing a multiengine sequence database search.

DOI: 10.1021/acs.jproteome.5b00850
J. Proteome Res. 2016, 15, 707−712

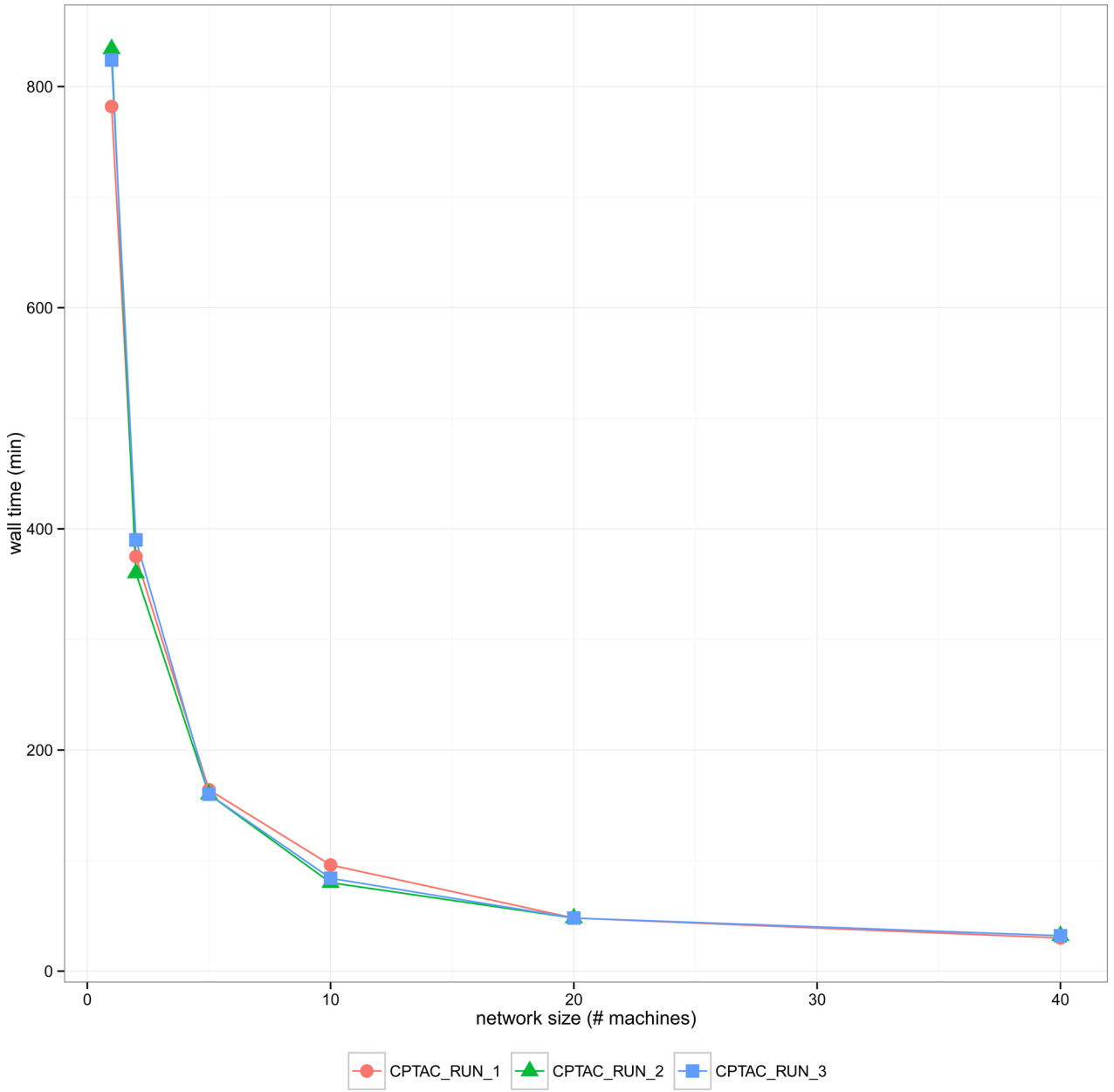

Figure 4. Size increase of the *Executer* pool causes a near exponential decrease in wall time, only limited by the time required for a single *Job* to finish and the implicit overhead due to network traffic. The experiment was repeated three times, showing a consistent trend in the decrease in the wall time for the analysis of 52 CPTAC runs using SearchGUI and PeptideShaker.

Thanks to its versatile structure, and the possibility to integrate third party software, Pladipus is not limited to proteomic approaches, as it allows the combination, execution, and management of a great variety of tools. It is thus particularly suited for computationally demanding multiomics strategies, where large data sets need to be processed with heterogeneous tools. Hence, it encourages scientists to seek out novel collaborations and collectively build, improve, and share pipelines beyond the boundaries of their field.

AUTHOR INFORMATION

Corresponding Author

*Tel: +32 9 264 93 58. Fax: +32 9 264 94 84. E-mail: lennart.martens@vib-ugent.be.

Notes

The authors declare no competing financial interest.

We thank Iain Buchanan for his technical support, help, and suggestions and Elodie Adam for the Pladipus logo design. Pladipus is freely available under the permissive Apache2 open source license at https://compomics.github.io/projects/pladipus.html.

ACKNOWLEDGMENTS

K.V. acknowledges Ghent University and VIB. L.M. acknowledges the support of Ghent University (Multidisciplinary Research Partnership "Bioinformatics: from nucleotides to networks"), Ghent University grant BOF12/GOA/014, and the IWT SBO grant "INSPECTOR" (120025). H.B. is supported by the Bergen Research Foundation.

DOI: 10.1021/acs.jproteome.5b00850
J. Proteome Res. 2016, 15, 707−712

REFERENCES

(1) Chandramouli, K.; Qian, P.-Y. Proteomics: challenges, techniques and possibilities to overcome biological sample complexity. *Hum. Genomics Proteomics* **2009**, *2009*, 1.

(2) Vizcaíno, J. A.; Côté, R. G.; Csordas, A.; Dianes, J. A.; Fabregat, A.; Foster, J. M.; Griss, J.; Alpi, E.; Birim, M.; Contell, J.; et al. The PRoteomics IDEntifications (PRIDE) database and associated tools: status in 2013. *Nucleic Acids Res.* **2013**, *41* (Database issue), D1063−D1069.

(3) Wilhelm, M.; Schlegl, J.; Hahne, H.; Gholami, A. M.; Lieberenz, M.; Savitski, M. M.; Ziegler, E.; Butzmann, L.; Gessulat, S.; Marx, H.; et al. Mass-spectrometry-based draft of the human proteome. *Nature* **2014**, *509* (7502), 582−587.

(4) Kim, M.-S.; Pinto, S. M.; Getnet, D.; Nirujogi, R. S.; Manda, S. S.; Chaerkady, R.; Madugundu, A. K.; Kelkar, D. S.; Isserlin, R.; Jain, S.; et al. A draft map of the human proteome. *Nature* **2014**, *509* (7502), 575−581.

(5) Verheggen, K.; Barsnes, H.; Martens, L. Distributed computing and data storage in proteomics: Many hands make light work, and a stronger memory. *Proteomics* **2014**, *14*, 367−377.

(6) Giardine, B.; Riemer, C.; Hardison, R. C.; Burhans, R.; Elnitski, L.; Shah, P.; Zhang, Y.; Blankenberg, D.; Albert, I.; Taylor, J.; et al. Galaxy: a platform for interactive large-scale genome analysis. *Genome Res.* **2005**, *15* (10), 1451−1455.

(7) Boekel, J.; Chilton, J. M.; Cooke, I. R.; Horvatovich, P. L.; Jagtap, P. D.; Käll, L.; Lehtiö, J.; Lukasse, P.; Moerland, P. D.; Griffin, T. J. Multi-omic data analysis using Galaxy. *Nat. Biotechnol.* **2015**, *33* (2), 137−139.

(8) Goecks, J.; Nekrutenko, A.; Taylor, J. Galaxy: a comprehensive approach for supporting accessible, reproducible, and transparent computational research in the life sciences. *Genome Biol.* **2010**, *11* (8), R86.

(9) Sheynkman, G. M.; Johnson, J. E.; Jagtap, P. D.; Shortreed, M. R.; Onsongo, G.; Frey, B. L.; Griffin, T. J.; Smith, L. M. Using Galaxy-P to leverage RNA-Seq for the discovery of novel protein variations. *BMC Genomics* **2014**, *15*, 703.

(10) Jagtap, P. D.; Blakely, A.; Murray, K.; Stewart, S.; Kooren, J.; Johnson, J. E.; Rhodus, N. L.; Rudney, J.; Griffin, T. J. Metaproteomic analysis using the Galaxy framework. *Proteomics* **2015**, *15*, 3553−3565.

(11) Cock, P. J. A.; Chilton, J. M.; Grüning, B.; Johnson, J. E.; Soranzo, N. NCBI BLAST+ integrated into Galaxy. *GigaScience* **2015**, *4* (1), 39.

(12) Baker, D. Centenary Award and Sir Frederick Gowland Hopkins Memorial Lecture. Protein folding, structure prediction and design. *Biochem. Soc. Trans.* **2014**, *42* (2), 225−229.

(13) Cooper, S.; Khatib, F.; Treuille, A.; Barbero, J.; Lee, J.; Beenen, M.; Leaver-Fay, A.; Baker, D.; Popović, Z.; Players, F. Predicting protein structures with a multiplayer online game. *Nature* **2010**, *466* (7307), 756−760.

(14) Kessner, D.; Chambers, M.; Burke, R.; Agus, D.; Mallick, P. ProteoWizard: open source software for rapid proteomics tools development. *Bioinformatics* **2008**, *24* (21), 2534−2536.

(15) The UniProt Consortium. Activities at the Universal Protein Resource (UniProt). *Nucleic Acids Res.* **2014**, *42* (Database issue), D191−D198.

(16) Vaudel, M.; Barsnes, H.; Berven, F. S.; Sickmann, A.; Martens, L. SearchGUI: An open-source graphical user interface for simultaneous OMSSA and X!Tandem searches. *Proteomics* **2011**, *11* (5), 996−999.

(17) Vaudel, M.; Burkhart, J. M.; Zahedi, R. P.; Oveland, E.; Berven, F. S.; Sickmann, A.; Martens, L.; Barsnes, H. PeptideShaker enables reanalysis of MS-derived proteomics data sets. *Nat. Biotechnol.* **2015**, *33* (1), 22−24.

(18) Muth, T.; Weilnböck, L.; Rapp, E.; Huber, C. G.; Martens, L.; Vaudel, M.; Barsnes, H. DeNovoGUI: an open source graphical user interface for de novo sequencing of tandem mass spectra. *J. Proteome Res.* **2014**, *13* (2), 1143−1146.

(19) Altschul, S. F.; Gish, W.; Miller, W.; Myers, E. W.; Lipman, D. J. Basic local alignment search tool. *J. Mol. Biol.* **1990**, *215* (3), 403−410.

(20) Paulovich, A. G.; Billheimer, D.; Ham, A.-J. L.; Vega-Montoto, L.; Rudnick, P. A.; Tabb, D. L.; Wang, P.; Blackman, R. K.; Bunk, D. M.; Cardasis, H. L.; et al. Interlaboratory study characterizing a yeast performance standard for benchmarking LC-MS platform performance. *Mol. Cell. Proteomics* **2010**, *9* (2), 242−254.

(21) Fenyö, D.; Beavis, R. C. A method for assessing the statistical significance of mass spectrometry-based protein identifications using general scoring schemes. *Anal. Chem.* **2003**, *75* (4), 768−774.

(22) Diament, B. J.; Noble, W. S. Faster SEQUEST searching for peptide identification from tandem mass spectra. *J. Proteome Res.* **2011**, *10* (9), 3871−3879.

(23) Kim, S.; Pevzner, P. A. MS-GF+ makes progress towards a universal database search tool for proteomics. *Nat. Commun.* **2014**, *5*, 5277.

(24) Barsnes, H.; Martens, L. Crowdsourcing in proteomics: public resources lead to better experiments. *Amino Acids* **2013**, *44* (4), 1129−1137.

DOI: 10.1021/acs.jproteome.5b00850
J. Proteome Res. 2016, 15, 707−712