MEDINFO 2017: Precision Healthcare through Informatics
A.V. Gundlapalli et al. (Eds.)

doi:10.3233/978-1-61499-830-3-375

From Bench to Bedside: A View on Bioinformatics Pipelines

Blanca Flores[a], Dirk Hose[b], Anja Seckinger[b], Petra Knaup[a], Matthias Ganzinger[a]

[a] *Institute of Medical Biometry and Informatics, Heidelberg University, Heidelberg,Germany,*
[b] *Department of Internal Medicine, Heidelberg University Hospital, Heidelberg, Germany*

Abstract

Although sequencing technology has become widely available in recent years, the steps in bioinformatics pipelines are time-consuming and barely standardized. New tools to improve individual steps in a pipeline are frequently published and configurations can be quickly adapted to use new versions. We performed case studies with a representative set of pipeline management tools using the GEP-R pipeline, and a qualitative study of different software packages covering relevant classes of software tools. We use a software toolset of R environment, Docker, KNIME, and BPEL to review our first aim of technical and organizational challenges. We propose snapshotting, documentation management, and a hybrid approach for our second aim of approaches to reproducibility. In order to have fully reproducible results derived from raw data, we think that it is necessary to archive biomedical analysis pipelines and their necessary software components.

Keywords:

Medical Informatics; Computational Biology; Software

Introduction

Sequencing and analysis of genomic data have become valuable technologies in both research and clinical settings. An effective and rapid analysis of data can support the identification of target genes and decision-making on diagnosis and treatment options [1]. Sequencing technology has become relatively cheap and widely available. In contrast, the steps of bioinformatics analysis following sequencing are still time-consuming and barely standardized. Typically, bioinformatics facilities develop and maintain their own chain of analytical steps, also known as an analytical pipeline. A variety of computer programs are available for each step. They have specific characteristics that make them more or less suitable for an analysis target. Each program used in each analytical step generates output data that can be used as input for the next program in the analytical pipeline. Since new tools for improving individual steps of a pipeline are frequently published, pipeline configurations can be quickly adapted to make use of the new versions.

Management of bioinformatics pipelines is facilitated by tools used for tasks such as the execution of serial and parallel steps and handling complex dependencies, varied software, data file types, and fixed and user-defined parameters [2]. Selection of tools in the pipeline depends on user requirement and implementation setting. Unfortunately, newer versions of packages can lead to unintended side effects in the analytical pipeline that may cause different results in calculations. Even small changes, like handling decimal places differently, can eventually lead to changes in classification of data or decision-making.

The problem of changing pipeline components is intensified by modular software packages such as the statistical software R [3]. Typically, a certain function is encapsulated into a package. However, packages often make use of functions provided by other packages, with each package having its own update cycle. As a consequence, complex packages have a complex dependency tree and are common in bioinformatics.

The differences in the pipeline implementation and frequent changes in tools and their versions affect the system´s stability, reproducibility, and validation. For example, a change in risk-classification software as mentioned above might lead to attribution to a different prognostic group for a patient. In such a case, problems occur when results are compared to those of other patients classified on different versions or if procedures are repeated on different versions. Consequently, technical and organizational steps are necessary to manage documentation, ensure version control, and validate pipeline configurations in a clinical setting.

Standardization and validation of the software tools are strong requirements for pipelines used in clinical implementation [4]. Thus, we propose two approaches for achieving reproducible pipelines: Snapshotting complete configurations, and documenting pipelines in detail.

Snapshotting

Taking a snapshot of a pipeline can be compared to freezing the whole runtime environment including all software components as they are installed at the time of taking the snapshot. The snapshot can be conserved for later reactivation or be transferred to another environment to replicate the pipeline. The environment can be built in a very interactive way, which makes it easy to implement even complex dependencies with heterogeneous version requirements. For example, the latest version of a certain package might depend on an older version of another package and hence it may not be possible to simply use the latest versions of all packages.

Pipeline documentation

Another way to conserve the pipeline configuration is to document the precise pipeline configuration. Preferably, this is done in a machine-readable way, so the pipeline can be rebuilt automatically. This approach relies on the software management capabilities of the programs used in the pipeline: they should support unattended installation methods in order to install the correct programs reliably. For modular software like R, functions have to be provided to extract the list of installed packages with versions and automatically reinstall the software from such a list.

A bioinformatics pipeline has been developed for a project called “clinically applicable, omics-based assessment of survival, side effects, and targets in multiple myeloma (CLIOMMICS)” during the research stage. It has further ma-

tured for use in clinical routine care. Multiple myeloma is a malignant disease characterized by molecular heterogeneity that can be assessed by gene expression profiling (GEP). The GEP-R is a reporting tool developed using R [3] and Bioconductor [5] and generates a medical letter summarizing the scores derived from the GEP analysis. Thus, it allows the automated interpretation of Affymetrix gene expression microarray profiles by using a bioinformatics pipeline. It aims to provide quality controlled, validated, and clinically digestible information that includes molecular classification, risk stratification, and assessment of target gene expression [2]. The result of the GEP-R bioinformatics pipeline is a report in the form of a letter to the physician in charge of patient treatment.

The aim of this paper is to describe the technical and organizational challenges of handling changes for bioinformatics pipelines, using the GEP-R project as an example. In addition, we propose two approaches to improve stability and reproducibility of the pipeline, as well as validity of its tools.

Methods

To investigate the possibilities for increasing reproducibility of bioinformatics pipelines we performed case studies with a representative set of pipeline management tools. We selected the software tools with the intention to cover the reproducibility strategies of *pipeline documentation* and *pipeline snapshot*. We do not aim to provide a comprehensive overview of pipeline management software found in Leipzig, and Curcin et al. [6, 7]; instead, we performed a qualitative study of different software packages covering relevant classes of software tools. These classes are:

- Package-based analysis software
- Virtualization software
- Graphical pipeline management software
- Document-oriented pipeline management software

We used the GEP-R pipeline as a test case.

GEP-R pipeline

The biomedical objectives of this pipeline are described in the original publication by Meissner et al. [2]. GEP-R is a typical pipeline for the analysis of DNA microarray data from a technical perspective. Specifically, .CEL files derived from Affymetrix U133 Plus 2.0 DNA microarrays are analyzed to classify stage, prognosis, and treatment options for patients with multiple myeloma. The pipeline consists of a series of Bioconductor functions to preprocess the raw microarray data and subsequently calculate relevant scores. In addition, quality control indicators are calculated. All results are combined and visualized in a format that can easily be interpreted by physicians by using the Business Intelligence and Reporting Tools (BIRT) framework [8]. The pipeline uses standard packages from R and Bioconductor repositories, and in addition, a modified non-standard version of a specific package is required. A workflow representation of the pipeline components is shown in Figure 1.

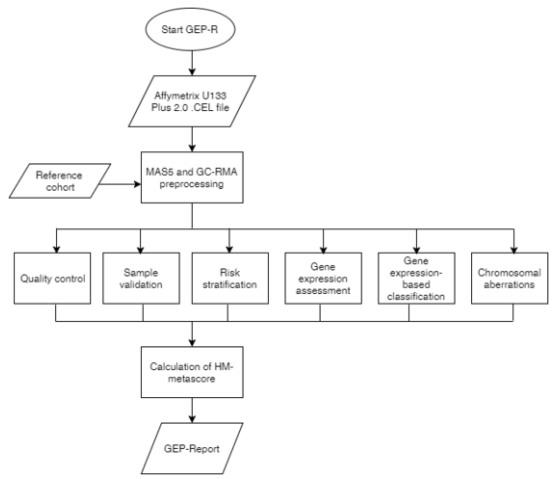

Figure 1- GEP-R pipeline

Software reviewed

Since R (often in combination with Bioconductor) is a very common pipeline component and at the same time a very complex system with complex internal dependencies, we consider it as a mandatory element of our test cases. Microsoft R repository is an approach for installing historic versions of R packages. It is possible to choose any date beginning 2014-09-17 within this repository and re-install a consistent R system based on package versions that were current on a specific date.

We chose KNIME Server version 4.3.2 [9] for a graphical pipeline management system for our test case. This tool has capabilities to interact with R and other external tools. The pipeline is manipulated as workflows via a graphical user interface. Workflows can be versioned and stored in a central repository.

We used Docker as a generic tool for preserving specific configurations of dependent software. With Docker, configurations can also be versioned and stored in repositories. In contrast to a pipeline management tool like KNIME, Docker cannot be used to provide a workflow by itself.

Finally, we investigated a document-oriented workflow management approach. BPEL can be used to design a workflow within the pipeline by describing its components with XML and Web Services. Changes in the pipeline can be tracked with version control and provide support for documentation management.

We performed our case studies on the basis of the bioinformatics pipeline established for the GEP-R report [2].

Results

The results are structured according to the research aims of our study. First, we describe the results for the evaluation of technical and organizational challenges. Second, we show our approaches for achieving reproducible pipelines by combining appropriate software components.

Aim 1: Technical and organizational challenges

R environment

R is a statistical software package that is very popular in biomedical research. However, its full scope cannot be leveraged by the monolithic core software, but relies on a huge amount of additional packages. R packages are typically developed by independent programmers and can be shared via the CRAN

network. As a result, release cycles of packages differ greatly and are typically not synchronized with the release cycle of core R. Since the behavior of packages can change during updates without notice, a newer version might break the pipeline.

The challenge of changing packages is attenuated to some extent by the R package checkpoint. This package was developed by Microsoft and allows rebuilding a consistent R environment using the Microsoft R Application Network (MRAN) repository. Unfortunately, it is not always the case that all packages used for a bioinformatics pipeline are at their current versions, since individual packages might have been added during the development process without updating the remaining packages.

In addition, bioinformatics applications are typically based on the Bioconductor tools. Bioconductor consists of more than 1200 R packages provided via its own repository system Bioc. Since Bioc is not included into the MRAN snapshots, it can be challenging if older versions of Bioconductor packages have to be installed manually.

Docker

Docker is an open source virtualization software that is installed on top of a variety of computer operating systems. It aims to virtualize individual applications in contrast to common virtualization approaches that work on complete computers. Docker provides mechanisms for easily moving the virtualized application (VA) between physical computers. The configuration of the VA can be versioned using the built-in snapshotting tool.

We successfully configured the complete R including Bioconductor and custom R packages inside a Docker container. This container can be copied to any computer that is intended for running our pipeline. Since the VA includes all required packages, we ensure consistent pipeline results across all instances. The integrity of the container is verified by comparing checksums of the VA's snapshot.

Graphical pipeline management system (KNIME)

KNIME is a general-purpose data analytics tool that provides a user interface for preparing a specific pipeline by graphically combining predefined components called nodes. For many analytical steps that typically occur in bioinformatics pipelines, nodes are shipped with KNIME or can be installed from an additional repository. However, not all specialized functions of Bioconductor are natively available in KNIME. Access to Bioconductor functions is available via KNIME's special R-nodes. These nodes are used for seamless integration of R-programs into KNIME workflows. Thus, data can be preprocessed in KNIME and R functions can be invoked as necessary within a workflow. The KNIME Server that we used for this test includes repository for storing and versioning pipeline configurations.

Document oriented pipeline management (BPEL)

The Business Process Execution Language for Web Services known as BPEL [10] is a process execution language standardized by OASIS. It allows the design of workflows using Web Services and represents data with XML specifications. Workflows can be linked and invoked by other Web Services, while inputs and outputs are assigned to variables to store data. Tools and technologies can be developed using Web Services to increase the level of automation in a process. Control structures used to manage tasks use either constructs that implement conditional branching and looping or activity containers to schedule sequential or parallel tasks. The capabilities of BPEL can be used for documentation management by describing the workflow of a pipeline with XML and tracking changes with its support for version control.

The GEP-R pipeline components can be described by expressing the exchanged data between the analytical steps as XML forms and by defining the order in which the Web Services are invoked. The steps in a process are known as activities and different methods can be used to manipulate data in the pipeline, such as invoking services, initializing variables, assigning values, and performing calculations. These can be combined into algorithms to perform complex processes. Changes in the pipeline can evolve and different versions can be deployed depending on the specific needs by using version control. Support for flexibility and adaptability to changes mitigate technical and organizational challenges and also simplify the process of documentation management.

Aim 2: Approaches for reproducibility

Snapshotting

Snapshotting seems to be the approach that is easier to implement since it is usually not disruptive to established development and implementation procedures. The granularity of scope that is covered by a snapshot depends on the software tools that are used. One approach is to cover complex components like R with Bioconductor. We established a Docker container that contained all specific configurations for our GEP-R pipeline. Since we were able to use the common interactive mode of installation, setting up the image for this container was relatively easy. Since the installation of packages is not limited to repositories, it was possible to compile and install a specifically modified R package into the container. When problems occurred during the installation process, it was possible to fix them without reinstalling the container as a whole. Since the latest snapshot covers any changes that were made to the container, it is hard to forget to document modifications as long as the snapshot is generated. Docker supports the development process by providing a versioning system that allows for multiple versions and even forks of images. Containers can be moved or copied to other computer systems or even organizations in order to reproduce a specific pipeline.

While it is easy to manage pipelines as a whole using snapshots, it can be quite complex for users to understand what is happening inside such a snapshot. For understanding the process, it is disadvantageous that the generation of the snapshot was done in an interactive way without enforcing to log all steps and programs that were involved. Essentially, a snapshot represents a black box with a behavior that can be reproduced very well but that can be complex to comprehend.

Documentation management

Documenting all pipeline steps and components leads to a description of the pipeline that makes it easier to understand its functionality. Placing the individual steps of a pipeline into a process description—for example using BPEL—can be achieved with reasonable effort. The situation with complex components or subsystems like R is more problematic. While it is possible to extract a list of all packages installed including the respective version, this list cannot be easily used for installing a new instance of R that exactly duplicates the first installation. Typically, only the latest versions available in the corresponding repositories of R and Bioconductor are used. For the installation of custom versions of R packages, individual installation procedures are required that go beyond documenting only package name and version.

For a fully automatic replication of a pipeline and its runtime environment, it is further necessary to use a documentation scheme that can be interpreted by a computer. Extensive testing of the document is necessary in order to ensure a complete

and error-free system. For each test, the whole pipeline environment has to be rebuilt from scratch. The whole pipeline development process has to be adapted to the documentation approach as a result.

Hybrid approach

We also analyzed a hybrid approach combining snapshotting and pipeline documentation for our tests based on the GEP-R pipeline. We implemented a Docker container only for the specific R environment in this case. The pipeline itself was managed and documented using the KNIME Server. The functional steps of the pipeline can easily be followed in the graphical notation with this approach. The black-box-effect of the Docker container is less significant since it is clear which specific functional step is called in the R environment.

Discussion

We assessed software tools and two different strategies for the documentation and archival of bioinformatics pipelines in order to facilitate reproducibility in our study. There are several factors that may influence the quality of bioinformatics pipelines: availability of a variety of tools for different steps, ability to handle changes in user requirements or tool versions. These factors can cause side effects such as small differences in results, but may have bigger impacts in terms of classifications and decision-making.

We considered three characteristics that are important for bioinformatics pipelines: stability, reproducibility, and ability to validate tools and versions. Changes in requirements and configurations may occur due to different reasons, such as frequent updates of tools, availability of new tools, and changes in requirements. A proper documentation strategy and version tracking allow smoother transitions and facilitate changes without disrupting the analytical steps or results.

We described two approaches for achieving these characteristics, each having its advantages and disadvantages. Snapshotting the whole pipeline is relatively easy to implement using virtualization technologies like Docker. However, it is not easy to follow the pipeline's functionality. It is also not known for how long container runtime environment will be available.

Documentation-based approaches provide pipelines that can easily be understood when looking at the whole process. However, they are more complex to implement. On the component level, they also face the problem that specific software components might not be available in the future.

A compromise of these two approaches is to document the high-level pipeline using tools like KNIME or BPEL. However, we suggest to use methods like Docker for individual components to conserve possible complex configurations for future use.

Research has been done to define formats for data expected to be used in future. More research is necessary to ensure the long-term availability of execution environments for the conserved pipelines.

Conclusion

Archiving raw data is considered good clinical practice as well as good research practice. However, in order for results derived from such data to be fully reproducible, it is necessary to archive biomedical analysis pipelines along with data as well as consider important requirements such as standardization and validation of all necessary software components.

Acknowledgements

CLIOMMICS is funded by the German Ministry of Education and Research within the e:Med initiative.
Grant id: 01ZX1609A.

References

[1] A. Valencia, M. Hidalgo, Getting personalized cancer genome analysis into the clinic: the challenges in bioinformatics. *Genome Med* **4** (2012), 61

[2] T. Meissner, A. Seckinger, T. Reme, T. Hielscher, T. Mohler, K. Neben et al., Gene expression profiling in multiple myeloma—reporting of entities, risk, and targets in clinical routine. *Clin Cancer Res* **17** (2011), 7240–7247

[3] R Development Core Team, *R: A Language and Environment for Statistical Computing.* Vienna, Austria, 2008

[4] ISO/IEC, Health software –Part 1: General requirements for product safety. Geneva, IEC, 2016. (vol 35.240.80) 2016. Available from: URL:http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=59543 [cited 2016 Dec 16]

[5] R.C. Gentleman, V.J. Carey, D.M. Bates, B. Bolstad, M. Dettling, S. Dudoit et al., Bioconductor: open software development for computational biology and bioinformatics. *Genome Biol* **5** (2004), R80

[6] J. Leipzig, A review of bioinformatic pipeline frameworks. *Brief Bioinform* (2016)

[7] V. Curcin, M. Ghanem, Scientific workflow systems - can one size fit all? In: 2008 Cairo International Biomedical Engineering Conference (CIBEC). p. 1–9

[8] D. Peh, N. Hague, J. Tatchell, *BIRT: A field guide. - Includes index.* 3rd ed., Addison-Wesley. Upper Saddle River, N.J., 2011. (Eclipse series)

[9] M.R. Berthold, N. Cebron, F. Dill, T.R. Gabriel, T. Kötter, T. Meinl et al., KNIME - the Konstanz information miner. *SIGKDD Explor* **11** (2009), 26

[10] Business Process Execution Language for Web Services., OASIS, 2007. Available from: URL:http://docs.oasis-open.org/wsbpel/2.0/wsbpel-v2.0.html

Address for correspondence

Matthias Ganzinger

Institute of Medical Biometry and Informatics

Im Neuenheimer Feld 130.3

69120 Heidelberg

Germany

Email: Matthias.ganzinger@med.uni-heidelberg.de