ProvBook: Provenance-based Semantic Enrichment of Interactive Notebooks for Reproducibility

Sheeba Samuel, Birgitta König-Ries

Heinz-Nixdorf Chair for Distributed Information Systems
Friedrich-Schiller University, Jena, Germany
sheeba.samuel@uni-jena.de, birgitta.koenig-ries@uni-jena.de

Abstract. With the rapid growth of data science and machine learning, interactive notebooks have gained widespread adoption among scientists across all disciplines to publish their computational experiments containing code, text, and results. As it is easy to modify and re-run the computations in a notebook, it is important to know how the provenance of results changed in different executions over the course of time, thus enabling trust and reproducibility. In this paper, we present ProvBook, an extension of Jupyter Notebook to capture and view the provenance over the course of time. It also allows the user to share a notebook along with its provenance in RDF and also convert it back to a notebook. We use the REPRODUCE-ME ontology extended from PROV-O and P-Plan to describe the provenance of a notebook. This helps the scientists to compare their previous results with the current ones, check whether the experiments produce the results as expected and query the sequence of executions using SPARQL. The notebook data in RDF can be used in combination with the experiments that used them and help to get a track of the complete path of the scientific experiments.

Keywords: Notebooks, Provenance, Reproducibility, RDF, Ontology

1 Introduction

The Jupyter Notebook [3] is an open-source web application to create documents with interactive output and supports over 40 programming languages with millions of users. Notebook documents contain blocks of text and code organized as cells. The code cells contain code snippets which can be modified and executed individually and the output is displayed directly below the cell. The markdown cells contain documentation of the computational processes. The cells are arranged linearly but can be moved or executed in any order. The notebook can be shared in different formats including HTML, PDF, and LaTeX.
Rule et al. [7] present a study in which they analyzed over 1 million publicly available notebooks and interviewed 15 data scientists from different disciplines. One of their results highlights the need for tracking provenance especially when

the cells are over-written and re-run. Provenance tracking is largely helpful in the trial and error experiments where it is essential to understand how exactly a final result has been achieved. It is also necessary to keep track of the experiments that have been attempted because that may benefit other scientists, even if the results are not as expected. Pimentel et al. [6] present a mechanism to capture and analyze provenance of python scripts inside IPythonNotebooks by integrating with noWorkflow [5]. PROV-O-Matic[1] is another provenance-tracking extension for older versions of IPython Notebooks which saves the provenance traces to Linked Data file using PROV-O. Another recent approach is to convert notebooks into workflows where notebook developers need to follow a set of guidelines in writing code [1]. These approaches have the limitation that they require changes to scripts by the user and are limited to Python scripts. In our approach, the provenance tracking is integrated within a notebook so there is no need to change the scripts and learn a new tool. It is also easy to share the notebook along with the provenance traces of execution described as Linked Data.

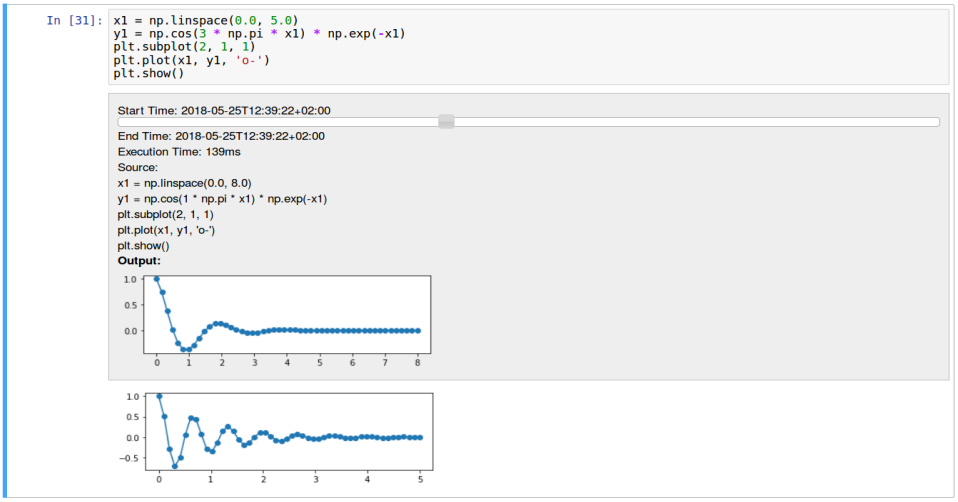

Fig. 1: A code cell with the provenance data of its executions.

2 Development

We developed ProvBook[2], an extension of Jupyter Notebook to capture and track the provenance of the cells over the course of the time. Every time the code cell is executed, the provenance of the run is stored in the metadata of the cell in the notebook. The provenance data of a code cell includes the start and

[1] `https://github.com/Data2Semantics/prov-o-matic`
[2] `https://github.com/Sheeba-Samuel/ProvBook`

end time of the execution, the total time it took to run the cell, the source and the output of the cell. It also allows the user to see the changes and the modified time of the text of a markdown cell whenever a notebook is saved. Figure. 1 shows the provenance of a code cell along with its input and output. When the extension is enabled, the visualization of the provenance data is displayed below the input of every cell. The user can view the history over the course of time by moving the slider. In this way, the user can compare the previous results with the current ones and see the difference that occurred. The user has the option to view the provenance of selected or all cells as well as clear them if needed. ProvBook also provides the user the ability to convert the notebooks to RDF. In our previous work [9], we have shown how we combined the P-Plan [2] and the REPRODUCE-ME [8] ontology extended from PROV-O [4] to describe the interactive notebooks along with the experiments which used them in a multi-user environment provided by JupyterHub. The metadata of the notebook fetched from JupyterHub API was stored along with the experimental data in a relational database and displayed in the project dashboard of our prototype. We extend our work by providing an extension to download the notebook as a Turtle document so that it is easy to share with the collaborators. The RDF file can also be converted back to the notebook for easy reading and execution. Figure. 2 shows how the provenance of the notebook is represented using

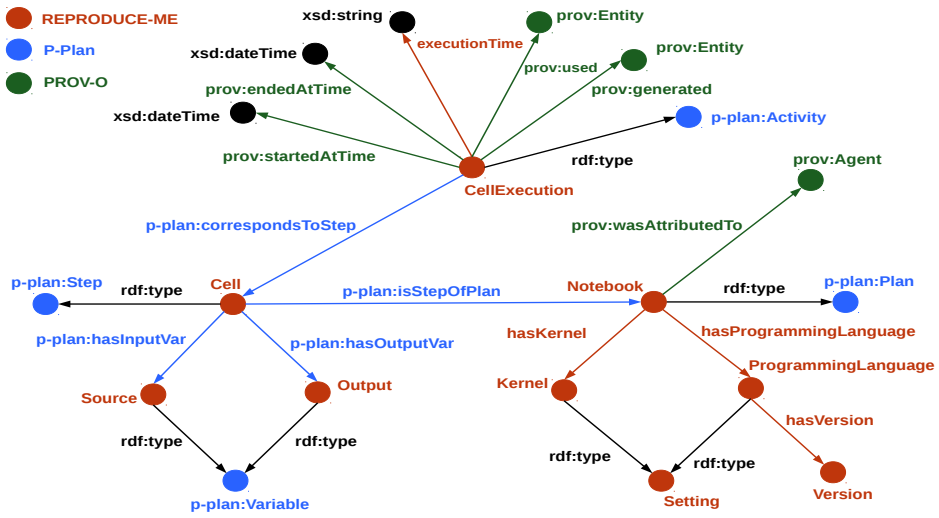

Fig. 2: The provenance of the notebook represented using the REPRODUCE-ME ontology.

the REPRODUCE-ME ontology. The notebook is described as a $p-plan{:}Plan$ with the cells as $p-plan{:}Step$ of the plan. Every run of a cell is represented as $CellExecution$, which is a subclass of $p-plan{:}Activity$. The $CellExecution$

uses input and generates output with start and end time. The notebook is attributed to the authors using the object property *prov:wasAttributedTo*. The provenance information including the kernel, programming language, and its version is available in the downloaded version of RDF. The ProvBook was evaluated with around 50 publicly available notebooks.

3 Demonstration Overview

We will demonstrate the working of ProvBook where the participants will be able to load or author notebooks, execute them and check the provenance of results. They will also be able to convert the notebooks to RDF and import them back to notebooks. We would also like to invite the participants to use their own notebooks with ProvBook. A video showing the installation and use of ProvBook with an example is available at `https://doi.org/10.6084/m9.figshare.6401096`.

Acknowledgements

This research is supported by the Deutsche Forschungsgemeinschaft (DFG) in Project Z2 of the CRC/TRR 166 High-end light microscopy elucidates membrane receptor function - ReceptorLight.

References

1. Carvalho, L.A.M.C., Wang, R., Gil, Y., Garijo, D.: Niw: Converting notebooks into workflows to capture dataflow and provenance (2017)
2. Garijo, D., Gil, Y.: Augmenting PROV with plans in P-Plan: scientific processes as linked data. CEUR Workshop Proceedings (2012)
3. Kluyver, T., Ragan-Kelley, B., et al.: Jupyter notebooks-a publishing format for reproducible computational workflows. In: ELPUB. pp. 87–90 (2016)
4. Lebo, T., Sahoo, S., McGuinness, D., Belhajjame, K., et al.: PROV-O: The PROV Ontology. W3C Recommendation 30 (2013)
5. Pimentel, J.a.F., Murta, L., Braganholo, V., Freire, J.: noWorkflow: A tool for collecting, analyzing, and managing provenance from python scripts. Proc. VLDB Endow. 10(12), 1841–1844 (Aug 2017)
6. Pimentel, J.F.N., Braganholo, V., Murta, L., Freire, J.: Collecting and analyzing provenance on interactive notebooks: When IPython meets noWorkflow. In: 7th USENIX Workshop on the Theory and Practice of Provenance (TaPP 15). USENIX Association, Edinburgh, Scotland (2015)
7. Rule, A., Tabard, A., Hollan, J.D.: Exploration and explanation in computational notebooks. In: Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems. pp. 32:1–32:12. CHI '18, ACM, New York, NY, USA (2018)
8. Samuel, S., König-Ries, B.: REPRODUCE-ME: ontology-based data access for reproducibility of microscopy experiments. In: The Semantic Web: ESWC 2017 Satellite Events, Portorož, Slovenia. pp. 17–20 (2017)
9. Samuel, S., König-Ries, B.: Combining p-plan and the reproduce-me ontology to achieve semantic enrichment of scientific experiments using interactive notebooks. In: The Semantic Web: ESWC 2018 Satellite Events. pp. 126–130 (2018)