1. Assessing and Restoring Reproducibility of Jupyter Notebooks

Jiawei Wang (1); Tzu-Yang Kuo (2); Li Li (1); Zeller, A. (3)
Source: *2020 35th IEEE/ACM International Conference on Automated Software Engineering (ASE)*, p 138-49, 2020; **ISBN-13:** 978-1-4503-6768-4; **DOI:** 10.1145/3324884.3416585; **Conference:** 2020 35th IEEE/ACM International Conference on Automated Software Engineering (ASE), 21-25 Sept. 2020, Melbourne, VIC, Australia; **Publisher:** IEEE Computer Society, Los Alamitos, CA, USA
Author affiliation: (1) Monash University, Faculty of Information Technology, Melbourne, VIC, Australia (2) Hong Kong University of Science and Technology, China (3) CISPA Helmholtz Center for Information Security, Germany
Abstract: Jupyter notebooks-documents that contain live code, equations, visualizations, and narrative text-now are among the most popular means to compute, present, discuss and disseminate scientific findings. In principle, Jupyter notebooks should easily allow to reproduce and extend scientific computations and their findings; but in practice, this is not the case. The individual code cells in Jupyter notebooks can be executed in any order, with identifier usages preceding their definitions and results preceding their computations. In a sample of 936 published notebooks that would be executable in principle, we found that 73% of them would not be reproducible with straightforward approaches, requiring humans to infer (and often guess) the order in which the authors created the cells. In this paper, we present an approach to (1) automatically satisfy dependencies between code cells to reconstruct possible execution orders of the cells; and (2) instrument code cells to mitigate the impact of non-reproducible statements (i.e., random functions) in Jupyter notebooks. Our Osiris prototype takes a notebook as input and outputs the possible execution schemes that reproduce the exact notebook results. In our sample, Osiris was able to reconstruct such schemes for 82.23% of all executable notebooks, which has more than three times better than the state-of-the-art; the resulting reordered code is valid program code and thus available for further testing and analysis. (0 refs)
Inspec controlled terms: data visualisation - human computer interaction - program compilers - program diagnostics - scientific information systems - text analysis
Uncontrolled terms: Jupyter notebooks-documents - scientific computations - individual code cells - 936 published notebooks - possible execution orders - instrument code cells - nonreproducible statements - exact notebook results - executable notebooks
Classification Code: C7330 Biology and medical computing - C1140Z Other topics in statistics - C6130 Data handling techniques - C6130B Graphics techniques - C6150C Compilers, interpreters and other processors - C6150G Diagnostic, testing, debugging and evaluating systems - C6180 User interfaces - C7210N Information networks
IPC Code: G06F7/00 - G06F11/36 - G06F8/41 - G16B
Treatment: Practical (PRA) - Theoretical or Mathematical (THR)
Database: Inspec
Data Provider: Engineering Village