SDEval: A Benchmarking Toolkit for Computer Algebra

Hans-Gert Gräbe¹  Albert Heinle²  Viktor Levandovskyy³

¹Institut für Informatik, Leipzig University, Germany
²David R. Cheriton School of Computer Science, University of Waterloo, Canada
³Lehrstuhl D für Mathematik, RWTH Aachen University, Germany

Observations
We observed the following in the Computer Algebra community:
- There are no standards to reliably reconstruct and verify computations in research articles.
- For a given computation problem in the computer algebra community, there is mostly no standard test set defined to examine the quality of a new implementation.
- A variety of Computer Algebra Systems (CAS) is available with distinct syntaxes and strengths.
- One CAS might fail to do a certain computation, while another one succeeds.
- Even though attempts are made to have a unified interface (e.g. SAGE), researchers still need to be accustomed to many CAS to get full access to the assistance available through CAS implementations.

Introducing SDEval
SDEval [3] is a benchmarking toolbox built on top of the SymbolicData database [1, 2]. Some of its main goals are:
- providing an easy way of translating existing entries in SymbolicData into executable code of computer algebra systems,
- providing a feasible way of trustfully reproducing and verifying computation results from current research papers,
- meeting the particularities of benchmarking in the field of computer algebra and
- being flexible in order to be applicable across different communities.

SDEval targets, among others, two groups of researchers:
- Those who want to try out different CAS to find a solution to a problem that appeared in his/her research.
- Those who have created a new implementation for a certain computer algebra problem and wish to compare it to existing ones.

The Taskfolder – An Easy Way to Reproduce Results.
- The taskfolder has the following structure:
  - Taskfolder
    - runTasks.py
    - taskinfo.xml
    - machinesettings.xml
    - classes
    - cashotcodes
  - Example translated into CAS code
  - Machine specific info

- The Taskfolder contains a script called runTasks.py. It can be run using optional parameters.
  Example:
  ```python
runtasks.py -c3600 -m1000000000 -j4
  ```

Furthermore, we have the following features:
- Visualization of the status as HTML files.
- User can manually terminate a CAS without having to restart the whole process.
- An interface for scripts interpreting the output of the CAS is given.
- The user can customize the computation by altering a preference-file given in XML format.
- Run-part independent from creation-part. One can write a taskfolder containing one’s own code and use our scripts to run and monitor them.

Directions for the Future
- Support more computation problems and CAS.
- Communication with different communities about further use-cases and feature-needs for SDEval.
- Provide meaningful output-interpretation scripts in the distribution of SDEval. This is challenging since:
- outputs coming from algorithms in Computer Algebra are often not unique and
- the evaluation of the correctness of an output is often not trivial and sometimes even subject of ongoing research.
- Preaching the practice of publishing taskfolders, so that computations in the literature can be verified.

References

Acknowledgements
We thank the “Deutsche Forschungsgesellschaft” (DFG) for partial financial support for the development of SDEval in the context of the "DFG Priority Project SPP 1489: Algorithmic and Experimental Methods in Algebra, Geometry and Number Theory.”

Resources
- [http://www.symbolicdata.org](http://www.symbolicdata.org)
- [https://github.com/ioah86/symbolicdata](https://github.com/ioah86/symbolicdata)
- [https://www.youtube.com/watch?v=CctmrfisZso](https://www.youtube.com/watch?v=CctmrfisZso)
- [https://github.com/ioah86/sage-benchmarking](https://github.com/ioah86/sage-benchmarking)
- [http://www.symbolicdata.org](http://www.symbolicdata.org)
- [https://www.youtube.com/watch?v=CctmrfisZso](https://www.youtube.com/watch?v=CctmrfisZso)