Automating Awkward Array Testing

Santan Roy Choudhury
(National Institute of Technology, Durgapur)

Mentored By: Ianna Osborne and Jim Pivarski
The Current Testing Infrastructure

create-tests.py

Generates a Python kernel from the kernel specs and generates the tests

Python Kernel
Kernel Specs

Python kernel tests
C kernel tests
CUDA kernel tests

Auto generated tests

Not a real kernel. Just a generated specification.
What are some of the loopholes here?

- Not too many specific test cases
- Not testing for specific errors
- The roles of the arguments are not well defined
Objectives of the Project

- Get more test data
- Store the data in a computer readable format (Yaml)
- Automate the process to generate tests from these data
The Test Data Generation Approach

Integration Tests

CPU Kernels

Data Passed to each Kernel Function

Calls the CPU Kernels at run time
Processing of the Raw Data

- Getting each arguments for each kernel function from the pool of raw data
- Getting the data for each argument correct and cleaning them of special characters or enormously large number
- Cleaning the data from unwanted numbers but keeping the length of the lists correct in the arguments.
- Mapping the processed data onto a yaml file.
The yaml Format

```yaml
unit-tests:
  - name: awkward_Identities_getitem_carry
    tests:
      - error: false
        inputs:
          carryptr: [0, 0, 0, 0]
          identitiesptr: [0, 1]
          lencarry: 4
          length: 2
          width: 1
        outputs:
          newidentitiesptr: [0, 0, 0, 0]
      - error: false
        inputs:
          carryptr: [0, 1, 1, 1]
          identitiesptr: [0, 1]
          lencarry: 4
          length: 2
          width: 1
        outputs:
          newidentitiesptr: [0, 1, 1, 1]
```
Separating Tests from Kernel Specifications

Role based Data In Kernel Specifications → Test Data Yaml File
The Results

- Data for 123 out of 143 Kernel Functions
- 2673 Unique Test Cases in Total
Future Development Plans

- Analyse the tests to look for bugs in the existing codebase.

- Use the tests for development of the CUDA Kernel (Test Driven Development)
Things I learned from the Project

- Best Practices for writing Unit Tests
- Got introduced to some python libraries - Pytest and Hypothesis
- Best Practices for preparing a presentation for a conference
- Gave a Talk at PyHep
  - https://youtu.be/dcuZqDCwpYM
Thank You!!

https://github.com/SantamRC

santamdev404@gmail.com