gui4dispel4py: A dispel4py graphical user interface on the Open Science Data Cloud

Steven Rapp*, Theano Stavrinos†, Melissa Bica‡, Race Clark§
*Florida International University, Miami, FL 33199
†University of California, Los Angeles, Los Angeles, CA 90095
‡University of Colorado, Boulder, Boulder, CO 80309
§Cooperative Institute for Mesoscale Meteorological Studies, University of Oklahoma, Norman, OK 73069

***Abstract*—dispel4py enables users uncomfortable with computer programming to link together simple third-party computer programs in useful ways. Here, we present a dispel4py graphical user interface on the Open Science Data Cloud.**

I. Introduction

An increasing number of scientific disciplines work with large amounts of data, or ”big data”. This big data often requires the use of high performance computing infrastructures in order to achieve research goals in a timely fashion. Successfully utilizing these sorts of infrastructures can be a daunting task for scientists inexperienced in scripting, as simple computer programs and scripts must almost always be significantly modified before properly running on high performance computers.

Researchers at the University of Edinburgh have developed dispel4py [1], a Python library that uses abstract workflows, or graphs, to complete data-intensive applications on distributed computing infrastructures. dispel4py helps scientists and researchers by doing the work necessary to get simple scripts and computer programs ready to run on distributed computer clusters. dispel4py is modular, so that bits and pieces of a workflow can be reordered, moved, reorganized, or reused depending on the needs of the researcher. The library also lifts from the researcher any worries about the properties on the distributed computing infrastructure on which the programs are to run.

When made available on the Open Science Data Cloud (OSDC) [2], dispel4py becomes even more powerful. OSDC will host a library of simple Python scripts uploaded by contributing researchers from all over the world. We present a graphical user interface (GUI) that allows for the import and use of these simple routines, as well as their organization into dispel4py workflows. The GUI also enables researchers to import their own Python routines and contribute to the OSDC library.

II. Data and Methods

A. dispel4py

In this project, dispel4py version 2014.08, as obtained from University of Edinburgh researchers at the 2015 OSDC PIRE (Partnerships for International Research and Education) workshop, is used to test gui4dispel4py. We demonstrate the utility of the proposed GUI by providing a library of five simple Python scripts that complete mathematical operations: addition, subtraction, multiplication, modulo, and exponentiation.

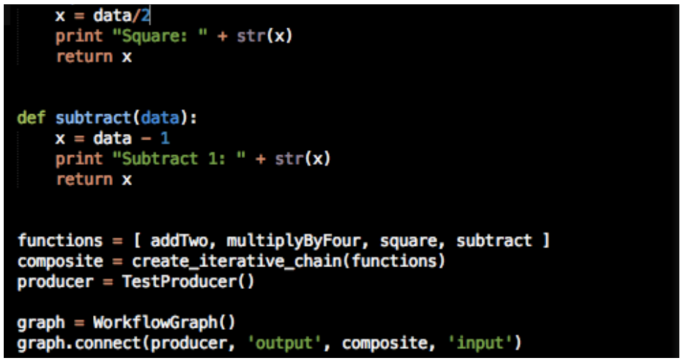
B. gui4dispel4py

gui4dispel4py is a web interface written in JavaScript and HTML. The major components of the interface are a ”toolbox” and a workflow. The toolbox contains Python scripts written by the user; these are the operations which the user wants to apply to their data and which, generally, would go in the body of the `process` of each dispel4py element class. The user can then pick up the elements in the toolbox and create a workflow by linking them together. After a suitable workflow is constructed, the user clicks “generate” and the GUI generates a Python file. This file contains the code necessary for running the user’s analysis on their data, i.e., it is the input to dispel4py, which the user will run from the command line.

III. Results and Discussion

We have demonstrated a GUI for dispel4py: gui4dispel4py. This GUI, in its first iteration, can successfully create, modify and run dispel4py workflows utilizing several simple Python scripts. gui4dispel4py consists of JavaScript, which underlies the user interface, organizes the constituent Python elements needed for the desired dispel4py workflow, and delivers a final Python script that serves as the input to dispel4py [3], as shown in Figure 1. The project also includes the HTML (HyperText Markup Language) code necessary to display gui4dispel4py in a Web browser. Figure 2 is an example of gui4dispel4py in use.

The gui4dispel4py interface is broken into two sections: ”Toolbox” and ”Workflow”. The Toolbox contains resources (simple Python scripts) that can be dragged and dropped into the Workflow. Users can choose from the default processing elements (PEs), represented by blue circles, in the toolbox or add their own via the ”Choose File” and ”Add” buttons at the top left of Figure 2. In this example, the PEs are named for mathematical operations they conduct, so ”Plus 2” adds 2 to the previous input. In the Workflow section of the interface, each box represents the position of a PE within the larger workflow. The individual PEs can be moved relative to one another to reorder the workflow. More than one PE can be



```
    x = data/2
    print "Square: " + str(x)
    return x

def subtract(data):
    x = data - 1
    print "Subtract 1: " + str(x)
    return x

functions = [ addTwo, multiplyByFour, square, subtract ]
composite = create_iterative_chain(functions)
producer = TestProducer()

graph = WorkflowGraph()
graph.connect(producer, 'output', composite, 'input')
```

Fig. 1: Example Python output from gui4dispel4py for input into dispel4py.

dragged onto the same box to create a compound element (CE), symbolized with a green circle, and in this case, named ”Compound 1” or ”Compound 2”. When the user is satisfied with the configuration of the dispel4py workflow, he or she can select ”Submit” and then the dispel4py routine as produced by gui4dispel4py will run, with the final output displayed over ”Output goes here” in Figure 2. In future work, the numeral displayed under ”Input” in the workflow portion of the interface would be replaced with a mechanism for the user to upload data on which the final dispel4py graph would operate.

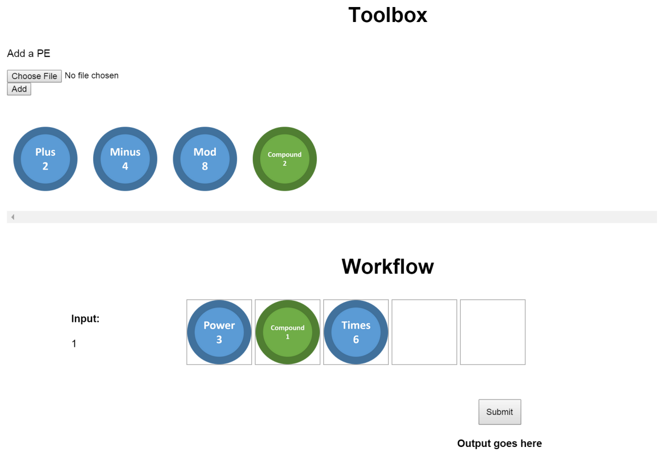


Fig. 2: The gui4dispel4py user interface.

IV. Summary

Researchers, scientists, and other OSDC users generally intimidated by the prospect of coding or by the prospect of transitioning from local computing to remote distributed computing will find a GUI for dispel4py useful.

The presented architecture could be easily extended to add important new features. We propose allowing users to import to their workflows any Python scripts from an OSDC-hosted library open to contributions from any interested OSDC user. Furthermore, gui4dispel4py users should be allowed to save their imported functions to the OSDC library for future use or to create composites of any already-extant functions in the library. Finally, dispel4py could be automatically launched as the final Python program is generated by gui4dispel4py.

Acknowledgment

This work made use of the Open Science Data Cloud (OSDC) which is an Open Cloud Consortium (OCC)-sponsored project. The OSDC is supported in part by grants from Gordon and Betty Moore Foundation and the National Science Foundation and major contributions from OCC members like the University of Chicago. Steven Rapp, Theano Stavrinos, Melissa Bica, and Race Clark were supported by the National Science Foundation Partnerships for Research and Education (PIRE) Award Number 1129076. Any opinions, findings, and conclusions or recommendations expressed are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

References

[1] R. Filguiera, I. Klampanos, A. Krause, M. David, A. Moreno, and M. Atkinson, “dispel4py: A python framework for data-intensive scientific computing,” in *Proceedings of the 2014 International Workshop on Data Intensive Scalable Computing Systems*. IEEE Press, 2014, pp. 9–16.

[2] R. L. Grossman, M. Greenway, A. P. Heath, R. Powell, R. D. Suarez, W. Wells, K. P. White, M. P. Atkinson, I. A. Klampanos, H. L. Alvarez, C. Harvey, and J. Mambretti, “The design of a community science cloud: The open science data cloud perspective,” in *SC Companion*. IEEE Computer Society, 2012, pp. 1051–1057.

[3] “gui4dispel4py github repository,” https://github.com/theanoli/gui4dispel4py, accessed: 2015-06-10.