Big Data Collaboration in Astrophysics: A View from the Trenches

Jeffrey P. Gardner
University of Washington
Box 351560, Seattle, WA 198195
jeffpg@uw.edu

ABSTRACT

Astrophysicists were among the first community to assimilate computing technology into their research. In many ways they continue to be at the leading edge, but in other ways some may consider them quite old-fashioned. I present an astronomer's view of the challenges of software design in a Big Data universe, discuss how Big Data is changing the ways in which astronomers design and use software, and present an argument for a possible solution that focuses on collaborative software development.

Author Keywords

Data-intensive computing; parallel computing; data-intensive collaboration; scientific collaboration.

SOFTWARE DESIGN IN A BIG DATA UNIVERSE

Astrophysics has been at the forefront in using both scalable computing technologies and data-intensive technologies. Computational astrophysics has used high-performance computing (HPC) platforms for decades to simulation the formation of structure in the Universe from the time of the Big Bang. The Sloan Digital Sky Survey[1] (SDSS) was the harbinger for the era of "data-intensive" science, where a large and complex dataset was made available to the entire community in a manner that was searchable and queryable through a variety of interfaces, including SQL. SDSS was transformational to the discipline, and its success has motivated many larger surveys such as Pan-STARRS[2] and LSST[3].

These successes notwithstanding, it is ironic that the manner by which astronomers interact with computers has remained largely unchanged for decades. I am an astronomer, and fully acknowledge my guilt in this regard. With the exception of some of the analysis environments I discuss in the next section, we usually write our own computer programs in general languages like C, Fortran, or Python using our favorite text editor, compiling and running them in a command-line environment. Although we have been aware of versioning systems for quite some time, most code is not written using one. Code re-use is low. Collaborative development is unusual. This behavior, I will argue, is the result of evolutionary pressures.

In the age-old struggle between flexibility and usability, we have nearly always erred on the side of the former. Until now, usability has been a luxury, whereas flexibility is a need. However, a third consideration has now firmly entered our design space: performance. Serial programs no longer offer sufficient performance to handle Big Data. Our analysis workflows must scale.

In order to do leading-edge science, we astronomers must find the right balance between flexibility, usability, and scalability. Fortunately, a subset of astrophysicists have been writing scalable HPC simulation codes for decades. Therefore, the skillset exists within the domain. However, the economics of simulation applications differ from those of data analysis. Simulation codes are used over many years by many people. Given this high degree of reuse, it is economically feasible to write such programs from scratch using low-level parallelism techniques such as message passing. But analysis codes come and go. Each astronomer has their own idea, their own physical or statistical model, i.e. their own mathematical workflow. This need for flexibility is what drove each of us astronomers to become our own software engineer in the first place, writing our own programs rather than using higher-level environments.

It is untenable, however, to imagine every astronomer writing their own message-passing codes from scratch in the same way they currently do in serial. Yet how do we maintain the same level of flexibility in software design that our research demands?

SOCIAL IMPACT OF BIG DATA

It is important, however, to remember that even so there are dozens of data analysis applications and environments in astronomy[4]. Most of these focus on image analysis, and some on more general data manipulation. Most astronomers do not completely rely on these, however. I would further argue that we will be finding such packages increasingly less relevant for analysis of Big Data.

Another impact of the Big Data revolution is less obvious but no less important. Again, we can study this effect by looking at astronomy. The "traditional way" of doing observational astronomy is that I, the astronomer, go to a telescope and use it to collect some data. I then take that

CSCW'12, February 11–15, 2012, Seattle, Washington, USA.

data, analyze it, then publish it before I share it with anyone else. In other words, what differentiates me from all the other astronomers in the world is that I have access to different data than they do.

In a world where every astronomer does a fairly similar set of tasks, but just with different data, it is easy to see the utility of general-purpose software environments. However, the trend in astronomy is to build larger sky surveys that will make their datasets available to everyone. Astronomers can no longer exclusive use access to data as a competitive advantage. At this point, it is just our capability as scientists that differentiate us from one another. In other words, *each of us has to do different math*. Consequently, any analysis environment that will be relevant in the Big Data universe is one that gives us the flexibility that we currently derive from using imperative general-purpose languages.

SCALABILITY THROUGH COLLABORATION

Thus far, I have presented two possibilities for scientific software development: “roll you own” vs. “pre-packed environments.” I have argued that the former is not tenable in a scalable Big Data world, and the latter is too restrictive. Somewhere between these two extremes lies a middle ground of ongoing collaborative development. Astronomers will always need to be software engineers of some sort. What we should be doing, however, is to figure out how to build environments that enable us to be better at collaborative development so that we each develop software components that each other can use and build upon.

This approach is not new. It is employed successfully in many large technology companies today. The key difference is that the company enjoys the ability to mandate software design standards and practices. It is harder for a community (one which is made of individual competitors) to agree on software design principles and practices. It is also difficult to envision training everyone in astronomy to become experienced software engineers. Researchers are, fundamentally, individual entrepreneurs who seek to maximize the return on their time. So far, there is low return-on-investment for sharing of software.

The goal, therefore, must be to design and build collaborative development environments that make it easier to do the right thing from a software design perspective. They must reward people for making their programs modular with APIs that are easy to understand. They must reward researchers for sharing their code with others. If sharing becomes both easier and higher-reward, then it will be possible to motivate the community to work together to build the scalable software that we all need.

In the 1990s, the Sloan Digital Sky Survey was a leap of faith on the part of astronomers. They believed if they worked together to produce a dataset that they all shared, the result would be better science for everybody. It is important to remember that it was not obvious at the time that this would indeed be the case. Now we must pose the similar question for software as we did for data. If we can take the leap of faith as a discipline and share the burden of building scalable software tools within a shared infrastructure for collaboration, we can all do bigger and better science.

To accomplish this, astronomers need to work closely with experts in the field of data-intensive collaboration. Consequently, I encourage these two communities to come together and share insights and ideas.

REFERENCES

1. http://www.sdss.org/.
2. http://pan-starrs.ifa.hawaii.edu/public/
3. http://www.lsst.org/
4. One of the most complete lists is available at http://www.atnf.csiro.au/computing/software/.