BIOINFORMATICS APPLICATIONS NOTE Vol. 28 no. 23 2012, pages 3139–3140
doi:10.1093/bioinformatics/bts573

Genome analysis

Advance Access publication September 28, 2012

Creating reusable tools from scripts: the Galaxy Tool Factory

Ross Lazarus[1,*], Antony Kaspi[1], Mark Ziemann[1] and The Galaxy Team

[1]Baker IDI Heart and Diabetes Institute, Melbourne, Victoria 3004, Australia

Associate Editor: Alex Bateman

ABSTRACT

Motivation: Galaxy is a software application supporting high-throughput biology analyses and work flows, available as a free on-line service or as source code for local deployment. New tools can be written to extend Galaxy, and these can be shared using public Galaxy Tool Shed (GTS) repositories, but converting even simple scripts into tools requires effort from a skilled developer.

Results: The Tool Factory is a novel Galaxy tool that automates the generation of all code needed to execute user-supplied scripts, and wraps them into new Galaxy tools for upload to a GTS, ready for review and installation through the Galaxy administrative interface.

Availability and implementation: The Galaxy administrative interface supports automated installation from the main GTS. Source code and support are available at the project website, https://bitbucket.org/fubar/galaxytoolfactory. The Tool Factory is implemented as an installable Galaxy tool.

Contact: ross.lazarus@channing.harvard.edu

Received on July 12, 2012; revised on September 7, 2012; accepted on September 17, 2012

1 INTRODUCTION

Galaxy (Blankenberg, 2010; Goecks, 2010; Nekrutenko and Taylor, 2012) is a web accessible application for high-throughput genomics, exposing popular third-party data sources and standard bioinformatics analysis packages in an integrated and persistent framework, designed to support biologist users performing reproducible analyses. There is a free public site (http://usegalaxy.org), and all Galaxy source code can be deployed locally (http://getgalaxy.org), where new tools can be created and installed to suit specific local requirements.

Rapid change seems to be one of the few constants in modern high-throughput biology so reporting and data transformation requirements also change rapidly. In groups where an established analysis framework such as Galaxy is used, software developers can create and expose new tools for local users, but even relatively simple tools require at least a few hours of skilled developer effort to implement, install and test. When a new simple transformation or report is needed, users will route around perceived damage, taking their data out, performing the required transformation with some quickly written code and then importing the transformed data back in to Galaxy for downstream analysis. Undesirable consequences are inevitable in terms of core scientific values such as reliability, repeatability and validity, when manual steps are performed outside an automated managed infrastructure.

*To whom correspondence should be addressed.

Many Galaxy users are capable of writing scripts to perform the required transformations, but lack the specific skills to convert these into Galaxy tools. This skill gap motivated us to build an automated method to run and test a user-supplied script inside Galaxy, and then to generate a new shareable Galaxy tool wrapping that script, requiring minimal specialized Galaxy skills and minutes rather than hours of developer effort once the script works correctly.

2 METHODS

Like many other Galaxy tools, the Galaxy Tool Factory (GTF) is implemented in Python. The required Galaxy tool wrapper descriptor is in XML as documented at http://bit.ly/Ui55jp. The GTF is run like other Galaxy tools, but instead of executing a standard bioinformatics analysis package, it calls an interpreter to execute a user-supplied script. Rscript, Perl, Python and shell scripts are currently supported, and extension to other interpreters is feasible. The GTF can only be executed by a local Galaxy administrator, whose login ID is listed in the 'admin_user' configuration parameter in universe_wsgi.ini, as it performs no security checks or sand boxing of the supplied script, as discussed later.

Each time the GTF is executed in Galaxy, the supplied script is run, and outputs are returned as a new persistent history item. This new result has the usual Galaxy 'redo' button, which conveniently recreates the GTF form complete with the script exactly as it was run, supporting rapid iterative script development through the standard Galaxy web interface. When the script works correctly, a complete new Galaxy tool can be generated, exposing the simple task defined by the small piece of now correctly working code (Box 1) pasted into the GTF form. Help text and other documentation to be shown to the user can also be pasted into the form, and are added to the generated tool as shown in Figure 1. The optional new tool is generated as a Galaxy Tool Shed (GTS, http://bit.ly/vwkB83) -compatible archive, ready to be uploaded as a new GTS repository.

Tasks are limited to being passed one history input file and one history output file, with no variable parameters. If multiple output files and images are written by the script, links to these can be automatically generated and presented to the user in an optional HTML history output.

The first and second command line parameters are used to pass the Galaxy input and output file paths, respectively, so the script must parse and deal with these correctly. An example in R that reads these paths is shown in Box 1. Working examples in Perl, Python and shell script are provided in the GTF documentation and on the tool form for testing and modification.

3 RESULTS

All of the routine coding required to convert a working script (e.g. Box 1) into a simple but fully integrated reusable and shareable Galaxy tool is automated by the GTF. A GTS-compatible compressed archive, containing the script, user interface and a functional test based on reproducing the supplied test case, is

© The Author 2012. Published by Oxford University Press. All rights reserved. For Permissions, please e-mail: journals.permissions@oup.com

Box 1. Example R script for a simple Tool Factory tool

```
# Write the transpose of a tabular input file
ourargs = commandArgs(T)
inf = ourargs[1]
outf = ourargs[2]
inp = read.table(inf, head=F, row.names=NULL, sep='\t')
outp = t(inp)
write.table(outp,outf, quote=F, sep='\t', row.names=F, col.names=F)
```

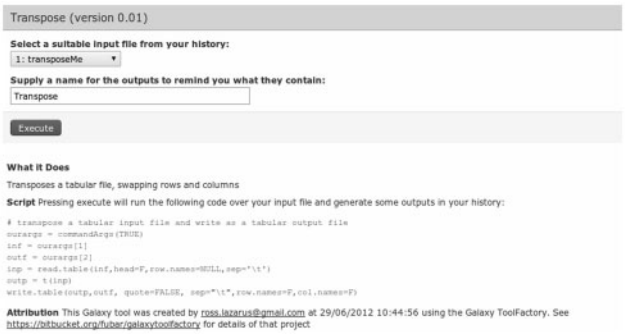

Fig. 1. User interface generated for script in Box 1

generated. GTS repositories can be searched, selected, reviewed and installed through the Galaxy administrative interface. GTF-generated tools perform identically to the same script wrapped by a skilled Galaxy developer. A developer can unpack the source code for a simple GTF-generated tool and edit it to add user-controlled parameters and new history outputs if needed, but alternative Galaxy tool interface generation methods described later may be preferred for complex requirements.

4 DISCUSSION

Complex computational pipelines for high-throughput data are often implemented as a sequence of relatively simple discrete steps, which typically read data from an input file, perform a transformation or calculation and then write an output file that serves as an input for a subsequent step. GTF-generated tools implement this simple model, ideal for creating simple components for work flows. The GTF currently supports popular scripting languages and makes it easy to test and debug them, and then generate complete simple work flow-compatible tools.

Only a local administrative user can actually execute the GTF because it performs no parsing or sand boxing of the supplied script, and therefore exposes insecure unrestricted scripting. It is recommended that users run it on private development clones, uploading tools to a GTS when they are ready for installation to production Galaxy sites. Installed GTF-generated tools run with normal Galaxy security, but administrators are urged to review all source code before installation.

The GTF creates complete Galaxy tools from scripts that perform simple transformation and reporting tasks. If multiple files and images are written, the HTML option automatically generates a complete web page of navigable links for the user.

The Bioconductor RGalaxy package can process R functions (http://www.bioconductor.org/packages/devel/bioc/html/RGalaxy.html) into Galaxy tool code. The CLI-mate (http://climate.lumc.nl/generate) resource generates Galaxy tool interfaces. Both require additional coding effort, and neither creates functional tests or integrates directly with Galaxy and the GTS. However, they can create complex tool interfaces with multiple user controllable parameters that are outside GTF scope.

Well-designed frameworks may help reduce risk and improve the repeatability, validity and efficiency of high-throughput biology analysis infrastructure for biologist users. Minimizing effort, and in turn, minimizing opportunities for users to introduce errors into analysis steps may help improve reliability. Standardized procedures can help maintain validity. Eliminating redundant coding effort will likely improve efficiency.

5 AVAILABILITY

The Galaxy administrative interface supports automated installation (https://bitbucket.org/galaxy/galaxy-central) from the GTS. Source code under an approved open source license and support are available at the project website, https://bitbucket.org/fubar/galaxytoolfactory.

6 CONCLUSION

The GTF executes user-supplied scripts in popular bioinformatics scripting languages and optionally turns them into reusable, shareable and inter-operable Galaxy tools, ready for work flows. GTF-generated tools run securely, but the GTF itself is not recommended for installation in production Galaxy instances. Generated tools can be uploaded and published through a GTS, which in turn supports automated tool installation into Galaxy servers. Once installed, when executed by a local user, GTF-generated tools pass a user-selected input file (Fig. 1) to the script provided at tool generation and return the output as a new history item. Supporting sharing and reuse of simple community-developed tools will help minimize the risks and costs of *ad hoc* local data transformation in high-throughput biology analyses.

Funding: The Galaxy Team is supported by NIH grants HG005133, HG005542, HG004909 and HG006620, as well as NSF grant DBI 0543285. Additional funding is provided, in part, under a grant with the Pennsylvania Department of Health using Tobacco Settlement Funds. The Department specifically disclaims responsibility for any analyses, interpretations or conclusions.

Conflict of Interest: None declared.

REFERENCES

Blankenberg,D. *et al.* (2010) Galaxy: a web-based genome analysis tool for experimentalists. *Curr. Protoc. Mol. Biol.*, Chapter 19, Unit 19.10.1–21.

Goecks,J. *et al.* (2010) A comprehensive approach for supporting accessible, reproducible, and transparent computational research in the life sciences. *Genome Biol.*, **11**, R86.

Nekrutenko,A. and Taylor,J. (2012) Next-generation sequencing data interpretation: enhancing reproducibility and accessibility. *Nat. Rev. Genet.*, **13**, 667–672.