g.remote

Remotely execute GRASS GIS scripts on your server

Vaclav Petras

Center for Geospatial Analytics
OSGeo Research and Education Laboratory
North Carolina State University

May 7, 2015

Server for everybody

- there are servers, HPC clusters, clouds lying around
- once somebody set it up, it's easy to get to it
  - if you know what `ssh -X` means
- and you also want to work locally in the same environment

Tangible Landscape

- currently locked to MS Windows desktop
- needs powerful processing backend for larger simulations
- pure in-cloud or client-server with WPS would be overkill

g.remote

- developed for hybrid desktop-server workflow
  - tests of processing or part of processing locally
  - store and process the big data on a server
- synchronous processing
- easily to integrate into scripts

Usage

Basic call in command line

```
g.remote user=john server=example.com \
    grassdata=/grassdata \
    location=nc_spm mapset=practice1 \
    grass_script=/path/to/script.py
```

- data are stored on the server
- Python script is local and transferred to the server

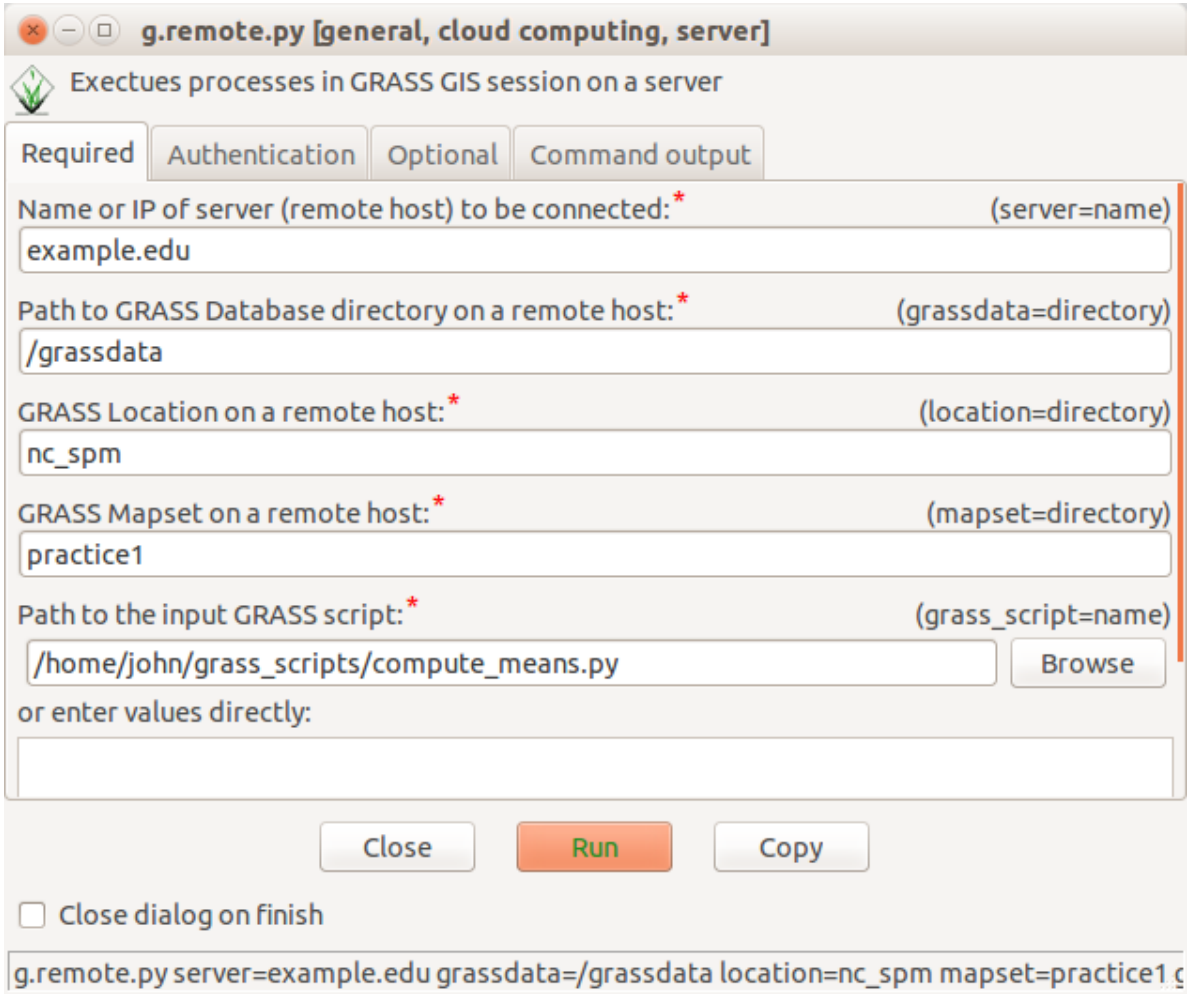
Usage

Addition of inputs and outputs

```
g.remote ... \
    raster=elevation \
    output_raster=waterflow
```

- data are transfered to and from the server

Usage


g.remote.py [general, cloud computing, server]
Exectues processes in GRASS GIS session on a server
Required
Authentication
Optional
Command output
Name or IP of server (remote host) to be connected:*
(server=name)
example.edu
Path to GRASS Database directory on a remote host:*
(grassdata=directory)
/grassdata
GRASS Location on a remote host:*
(location=directory)
nc_spm
GRASS Mapset on a remote host:*
(mapset=directory)
practice1
Path to the input GRASS script:*
(grass_script=name)
/home/john/grass_scripts/compute_means.py
Browse
or enter values directly:
Close
Run
Copy
Close dialog on finish
g.remote.py server=example.edu grassdata=/grassdata location=nc_spm mapset=practice1

Architecture

Three layers

- connection to server (class)
  - Paramiko
  - `ssh` + `scp` (OpenSSH Client)
  - *can accommodate web-based applications or local programs*
- GRASS session (class)
  - runs GRASS modules, scripts and Python code inside GRASS session using the connection
  - transports GRASS data (maps, region, . . .)
  - *independent on connection type*
- GRASS module (user interface)
  - specialized for a given task
  - *different modules can be implemented*

Server side

SSH server

- Linux (or anything unix-like)
- GRASS GIS
- OpenSSH Server

Docker

Like virtual machine (e.g. VirtualBox)

- processes contained inside
- shareable

Better than virtual machine

- fast (no overhead)
- easily shared and customized – configuration is a text file
- can contain data and applications or just one application

- but there is also Vagrant

Availability of g.remote

Source code and manual on GitHub

- `https://github.com/wenzeslaus/g.remote`

License

- GNU General Public License >=2.0

GRASS GIS Addons repository

- will be moved there after discussion with community

Software acknowledgment

- GRASS GIS
- OpenSSH, Paramiko, Python, Linux, Docker and a lot of other free and open source software
- Presentation done with LaTeX, Beamer and Overleaf

Thanks for your attention