UFR de mathématique et d'informatique
Université de Strasbourg

git-annex : a short introduction

Pierre-Antoine Senger

April 7, 2025

Overview

Context

Context

- **HPC** often requires large datasets (meshes, post-processing data, etc.).
- **Git** is not designed for large files and gets very slow.

Figure 1: Gaya HPC cluster

Figure 2: Git logo

Possible solutions

Keep the data locally

Works but not ideal, especially for:

- **Collaboration** (multiple users).
- **Reproducibility** (multiple runs).
- **Continuous integration** (CI).
- **Versioning** (multiple versions).
- **Backup** (multiple copies).

Figure 3: Laptop to keep the data locally

Avoidance

- **File generation** on demand or at execution time.
- **Data reduction:** e.g. only store the most meaningful data.
- **User specifies** which statistics/visualization to generate at each execution.
- *etc.*

Cloud storage

For everyday user: Google Drive, Dropbox, OneDrive, etc.
Targeted at scientific collaboration: Zenodo, Figshare, Dryad, OSF, etc.

Problematics:

- **Requires lots of scripting** to integrate in the HPC workflow.
- **Requires special security measures** to protect sensitive data.
- **How to detect and manage errors?**
- **Compatibility issues.**

Git LFS

Open source extension to Git.

Git Large File Storage

- Replaces large files with text **pointers** inside Git, while storing the file contents on a remote server.
- **No need for custom scripting.**
- **Consistent** between local and remote.

Suboptimal usage of local storage:

- **Uses locally twice the space** (files are duplicated in `.git/lfs`).
- **Large files** are **automatically downloaded** when cloning a repository.
- End users have **nearly no permission/control** on the remote server.

Git-annex

Open source extension to Git.

- Allows managing large files with Git without checking the file contents into Git.
- Uses **symlinks** to optimize local storage.
- **No duplication** of files.
- No intrinsic limit on file size or bandwidth.

More controls, the user can:

- decide at anytime which files to keep locally.
- use special command to **download** or **drop** files.

Git-annex

Supports the **download of large files** content from either:

- some **other git-annex repository** on another machine (if an ssh connection is possible).
- a **cloud storage provider** (e.g. Amazon S3, Google Drive, Dropbox, etc.)

Main drawback:

- **Non natively supported** by GitHub, GitLab, etc. (files needs to be managed with command lines).
- **Learning curve**.
- **Not as common** as Git LFS, support may be harder to find.

Installation and usage

Installation

- `sudo apt-get install git-annex` (Debian/Ubuntu)
- `sudo pacman -Syu git-annex` (Arch)
- `brew install git-annex` (MacOS)

Usage

Initialize and add a large file (single quotes are important):

```
git annex init ’PA laptop’
git annex addurl --file=large_file.zip download_url_link
# no need to git add
git commit -m "Add large_file.zip"
git push origin main git-annex
git annex list
git annex whereis large_file.zip
```

Usage

Retrieve a file from another repository:

```
git annex init ’Alice laptop’
git annex get .
```

Annex a newer version:

```
git annex drop large_file.zip
git rm large_file.zip
git annex addurl --no-check-gitignore --file=large_file.zip
git commit -m "Update large_file.zip"
git annex sync
```

Usage

If another user wants to retrieve the newer version:

```
git annex sync
# Initialize the download :
git annex get .
```

Check for old orphan symlinks files and their references number:

```
git annex unused
git annex dropunused NUMBER
```

The end

Thank you for your attention!

🎉

Any questions?

Contact Information:
pierre.antoine.senger@gmail.com
github.com/PA-Senger