git-annex

or: How I learned to stop worrying and manage the data

UNIVERSITY OF
TORONTO

Peter Colberg

University of Toronto

September 11th, 2013

scientific data trivia

- How do you annotate a collection of data?
  - e.g., readme files per directory, ...
  - When and where was the data recorded?
  - With what intention was the data produced?
- How do you synchronize data between machines?
  - e.g., scp, sftp, rsync, find, ...
  - Which data should be transferred?
  - Which data has been transferred already?
  - Which data is still being recorded?
- How do you duplicate your data?
  - e.g., cross fingers that nothing bad will happen to that one copy, ...
  - How do you keep track of where which data is?
  - Is all your data still accessible when a storage system fails?
  - How do you verify the integrity of your data?

git?

- git – the stupid **content tracker**
  - keep data in a git repository
  - annotate data using git commit and git log
  - synchronize data using git push and git pull
  - merge repositories using git fetch and git merge
  - ensure integrity using git fsck

- git not (yet?) useful for large files
  - two copies of data, in working tree and in object store
  - no partial checkout of specific files
  - repository contains the complete history
  - removed files still present in object store

use git to manage file *meta*data, not file contents

git-annex!

git-annex tracks files with git, without storing their contents into git

- `git annex add`
  - moves file within git annex objects directory
  - names file according to its content hash (e.g. SHA256)
  - makes file read-only
  - creates symbolic link to the file with original filename
- `git commit`
  - commits symbolic link to file, not file contents
- `git annex get`
  - transfers file contents from other repositories
  - uses `rsync`/`wget`/...over ssh/http/...

first commands

- git-annex is available for many OS, and on the SciNet gpc cluster

```
module load git-annex
```

- creating a git annex repository

```
cd data/
git init
git annex init
```

- adding files

```
git annex add
git annex add 2013/08/25/dimer/*/*.h5
git commit
```

- cloning a remote repository

```
git clone --origin scinet login.scinet...:/.../data
cd data/
git annex get
```

more commands

- transparently working with symbolic links to files

```
ls -L ...
cp -L ...
du -L ...
```

- adding a remote to an existing repository

```
git remote add scinet login.scinet...:/.../data
```

- synchronizing metadata between local and remote repositories

```
git annex sync
```

- pulling and pushing data from and to remote repositories

```
git annex get
git annex copy -t scinet
```

- updating working tree while logged in on remote host

```
git checkout -f master
```

even more commands

- checking the integrity of annexed files

```
git annex fsck
```

- dropping the contents of a file, given sufficient number of copies

```
git annex drop FILE
```

- increasing the required number of copies

```
echo "* annex.numcopies=3" >> .gitattributes
```

- untrusting a remote repository

```
git annex untrust devnullnet
```

- querying the locations of copies of a file

```
git annex whereis FILE
```

sufficiently many commands

- viewing history of data

```
git log --stat --decorate
```

- graphically viewing history of data graphically

```
gitk
```

- editing a file

```
git annex unlock FILE
git annex add FILE
```

- removing a file

```
rm FILE
git commit -a
```

- deleting the contents of removed files

```
git annex unused
git annex dropunused ...
```

caveats

- git-annex invokes separate rsync process per file
  - ssh connection shared between successive rsync calls
  - may not work due to socket path length restriction
- opportunistic connection sharing (OpenSSH 5.6 or newer)

```
# ~/.ssh/config
ControlPath ~/.ssh/master-%L-%r@%h:%p
Host host.example
  ControlMaster auto
  ControlPersist 60
```

- manual connection sharing

```
# ~/.ssh/config
ControlPath ~/.ssh/master-%L-%r@%h:%p
```

```
ssh -fNM host.example
```

caveats

- git-annex-shell needs to be in PATH on remote host

```
# ~/.bashrc
export PATH="/scinet/gpc/tools/git-annex/.../bin:$PATH"
```

- use SciNet's datamover1 node for transfers >10 Gb

```
module load hpnssh
```

- dereferencing hundreds of symbolic links may be slow
  - parallel file storage is designed for high throughput
  - not so much for low latency, or high number of file accesses
- git-annex provides a *direct mode*
  - when symbolic links are not available, or not an option
  - no safety net against command-line fu
  - limited subset of safely usable git (not git annex) commands

http://git-annex.branchable.com/