MATH 676

–

Finite element methods in scientific computing

Wolfgang Bangerth, Texas A&M University

Lecture 32.75:

Learning to use modern tools, part 5b:

Version control systems (VCSs)

Git

Git

Git has a different philosophy:

- There is a repository somewhere
- I create a *local copy* (“clone”) of it
- I check out from and into my own clone

- I can “pull” further changes from upstream (or another repository)
- I can “push” my locally committed changes into another repository

Git

Advantages:

- My own repository is like a branch
- I can work offline
- I can mix and match changes from different repositories
- Branches are more “natural” in git than in subversion and are used far more often
- It seems to somehow work better with inexperienced developers

Disadvantages:

- More difficult model (but there are many online tutorials)

Git

Terminology:

- Subversion “mainline” –> git “master”
- “origin” of repository *B*: by convention, the location of the repository *A* from which *B* was cloned
- “upstream” of repository *B*: by convention, the location from which *A* itself was cloned

Git

Terminology:

- In subversion one thinks of version *N* as the "state" of the code after *N* revisions have been made
- A commit results in a new version

- In git one tries to avoid thinking of versions; rather, as a collection of *patches* (i.e., "patch sets")
- A commit results in a new patch to be added to the current branch (e.g. "master")

Git

Conceptual differences: Revisions numbers etc.

- In subversion, there is only one repository
- We can enumerate all versions uniquely

- In git, there are many repositories
- None is superior to the others

  (though projects designate an “official” repository)

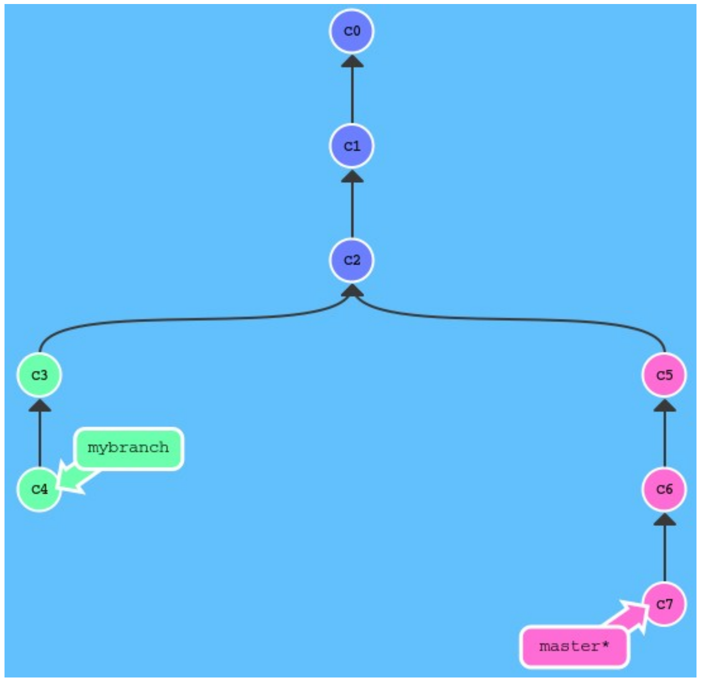
- Changes are moved from repository to repository

- Revision numbers would no longer match between repositories -> revisions are identified by hashes, versions are just patch sets

Git

Example for revision numbers/hashes:

- In the beginning:



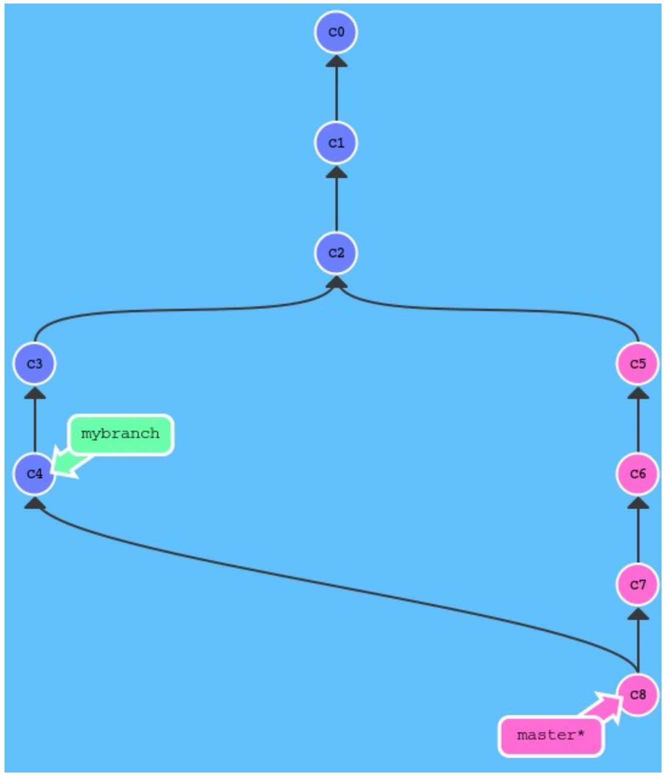
dealii.org

Using http://pcottle.github.io/learnGitBranching

Git

Example for revision numbers/hashes:

- deal.II developers make another change on their *master*:



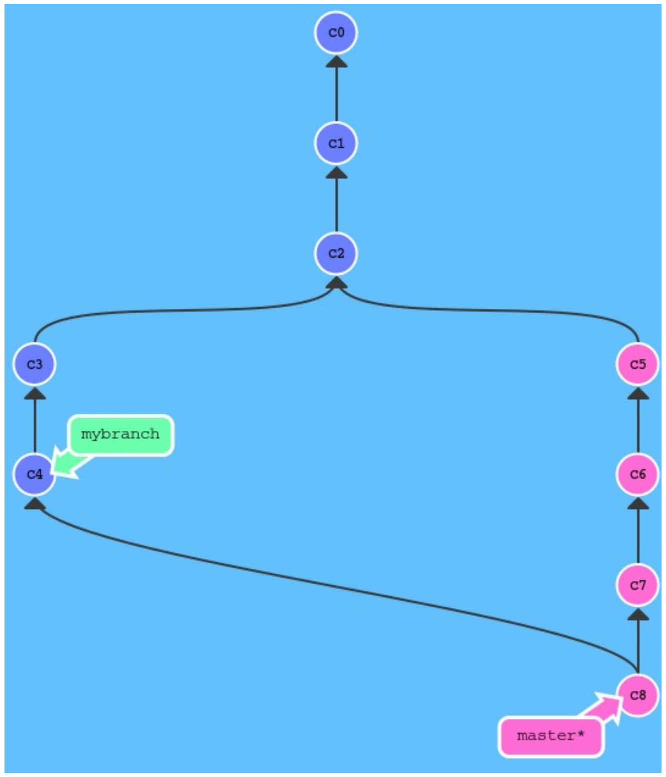
dealii.org

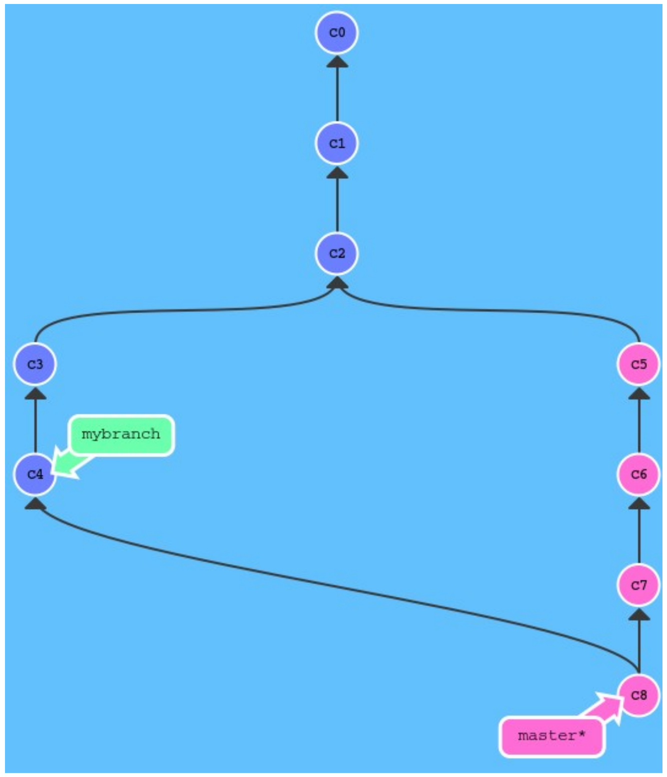
Using http://pcottle.github.io/learnGitBranching

Git

Example for revision numbers/hashes:

- I "clone" the repository to my local harddrive and check out a copy:





dealii.org

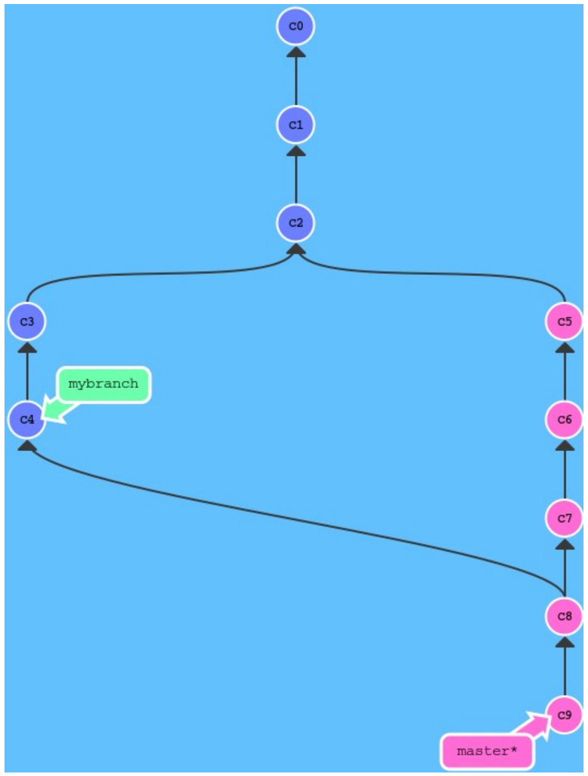
My clone

Using http://pcottle.github.io/learnGitBranching

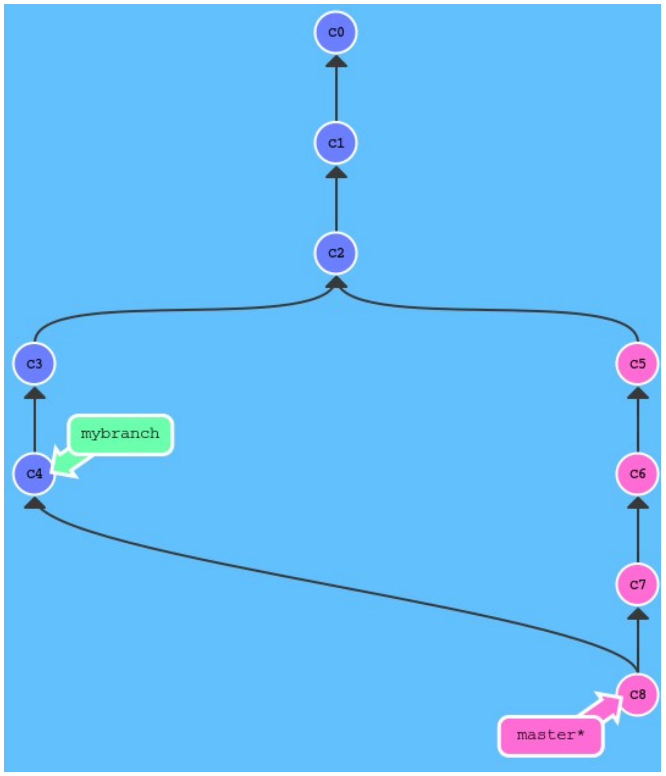
Git

Example for revision numbers/hashes:

- Somebody commits further changes to “origin”:



dealii.org



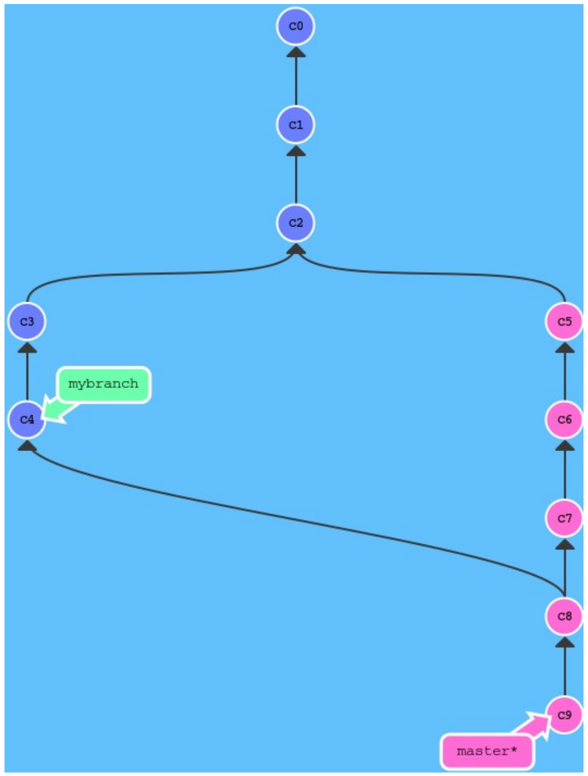
My clone

Using http://pcottle.github.io/learnGitBranching

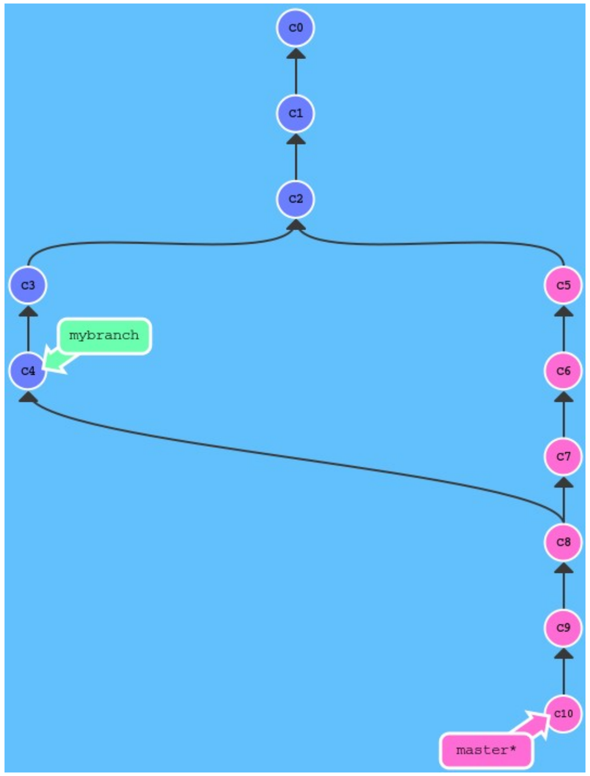
Git

Example for revision numbers/hashes:

- I make and commit two changes to my local clone:



dealii.org



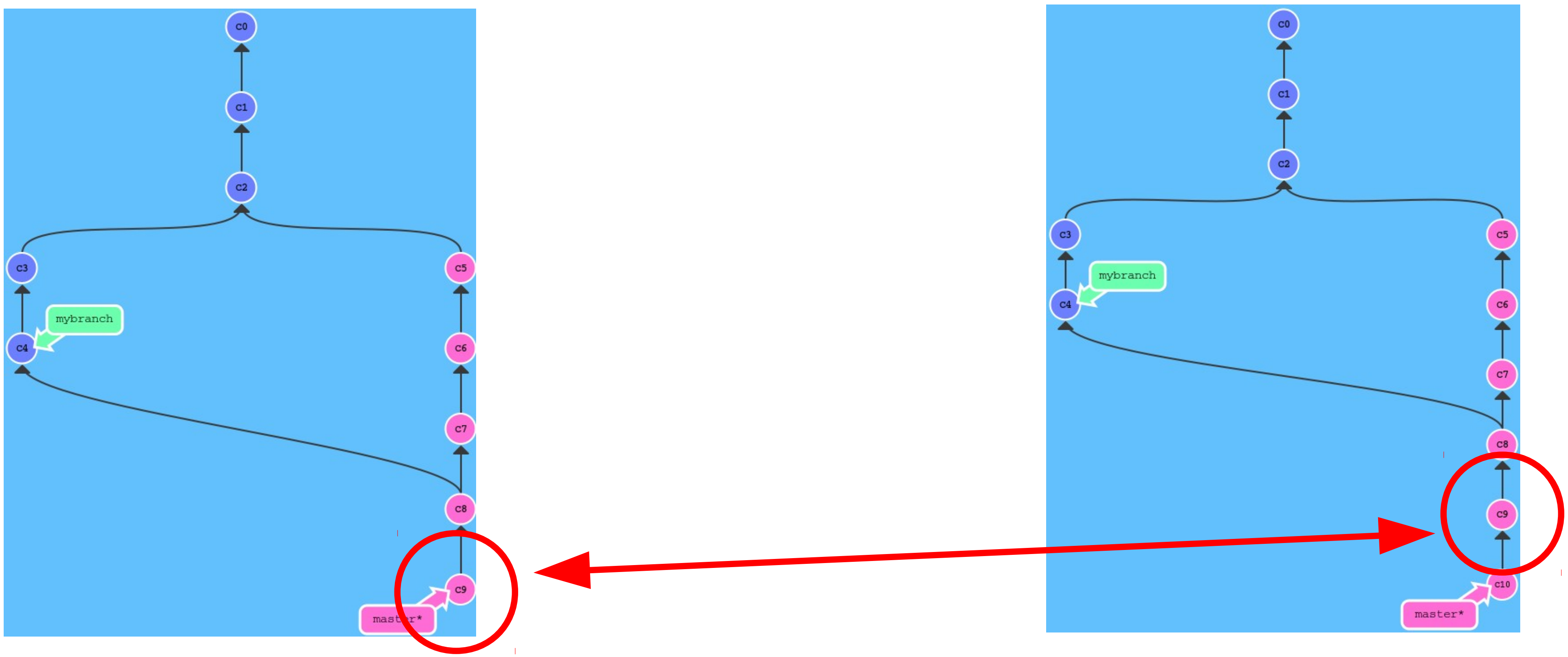
My clone

Using http://pcottle.github.io/learnGitBranching

Git

Example for revision numbers/hashes:

- **Note:** c9 in the two repositories are *different* changes!



dealii.org

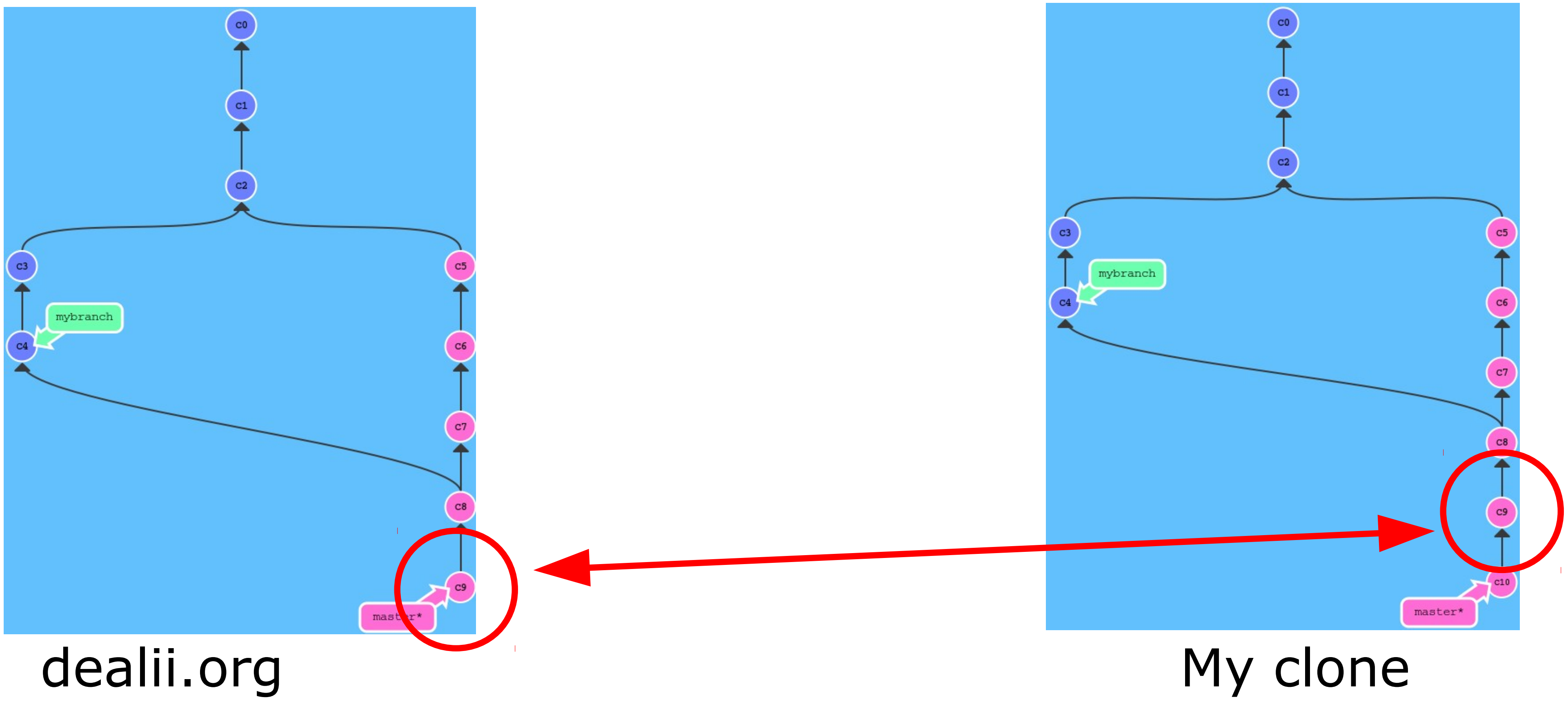
My clone

Using http://pcottle.github.io/learnGitBranching

Git

Example for revision numbers/hashes:

- **Consequence:** If I want to contribute c9 and c10 to "origin"(=dealii.org) they would have to get new numbers



Using http://pcottle.github.io/learnGitBranching

Git

The revision number problem:

- (Sequential) Numbers cannot be unique between repositories if no repository is "special"

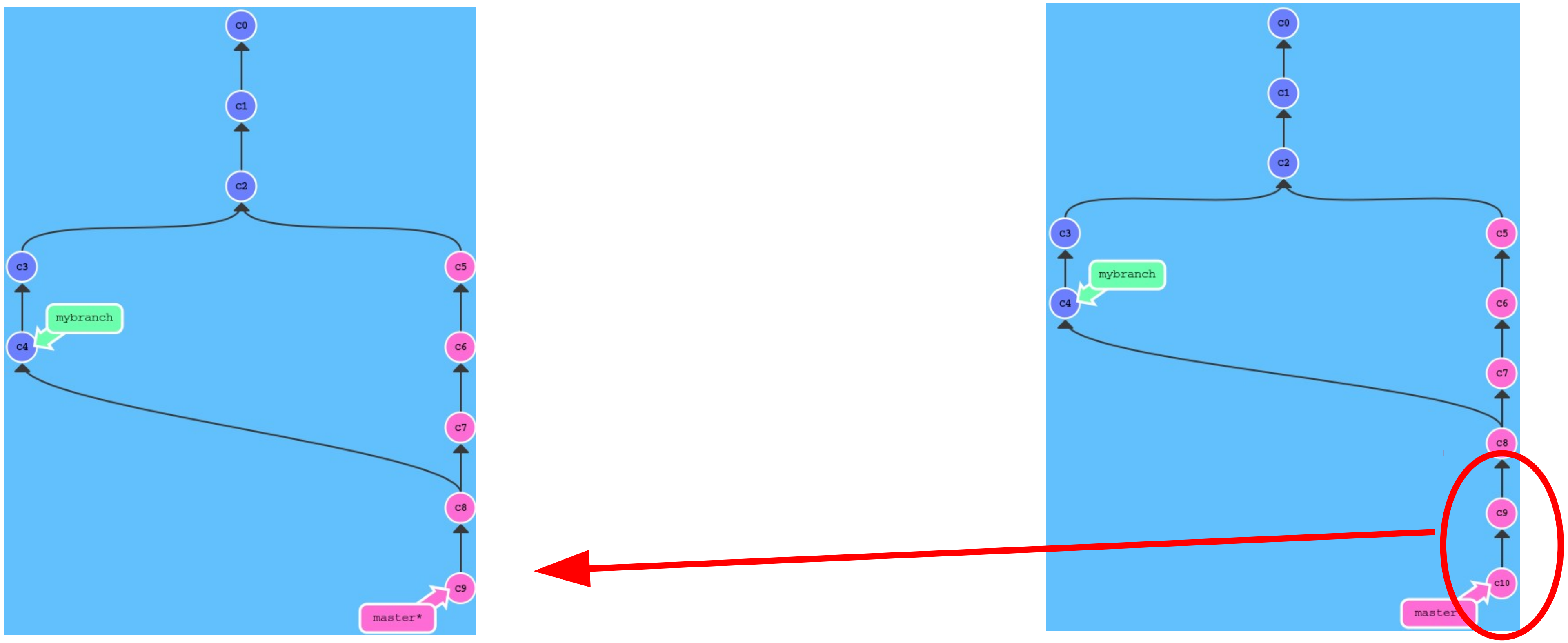
- Patches are identified by a 40-digit "hash", e.g.:
  525b8df065394c541d499cd6d62cf9485ec5038b
- Most of the time, we refer to patches only by their first 7 digits:
  525b8df

- A hash is computed from:
  – the ancestor of a patch
  – its content
- Hashes could conflict, but this never happens in reality

Git

Example for revision numbers/hashes:

- I want to contribute ("push") my changes:



dealii.org

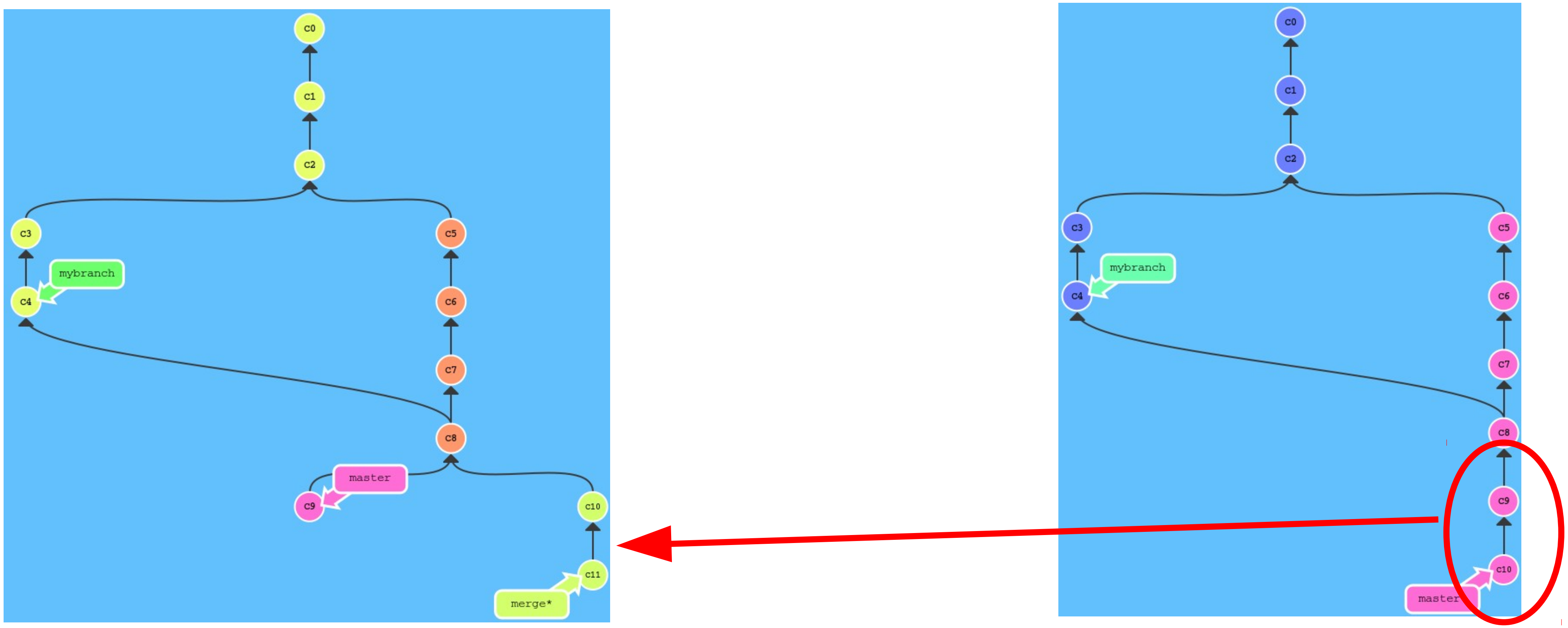
My clone

Using http://pcottle.github.io/learnGitBranching

Git

Example for revision numbers/hashes:

- Step 1: Attach the two patches to their ancestor (c8)



dealii.org

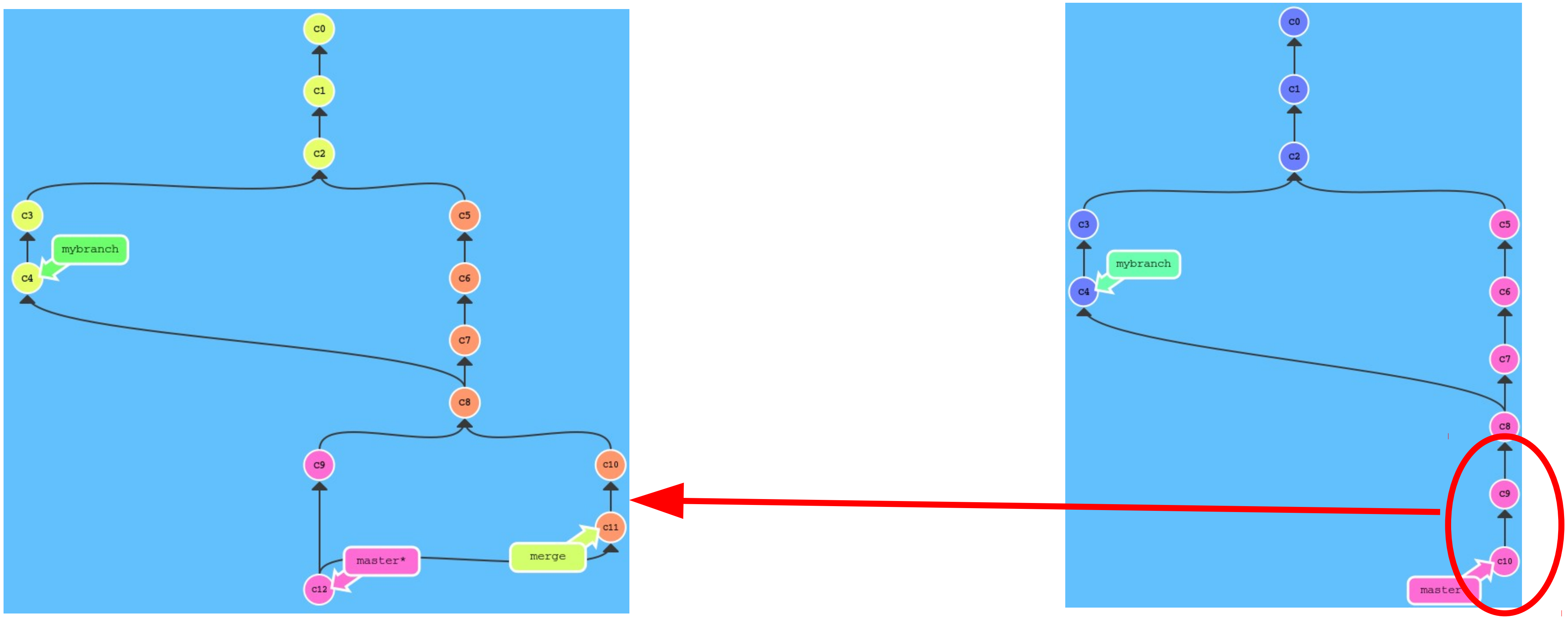
My clone

Using http://pcottle.github.io/learnGitBranching

Git

Example for revision numbers/hashes:

- Step 2: Merge the two changes past c9 to yield c12



dealii.org

My clone

Using http://pcottle.github.io/learnGitBranching

Git

Let's see some of this in practice...

Git

Conceptual differences: The sanctity of the repository

- In subversion, you never ever change anything that's been committed before
- In git, we do this all the time:
  - Delete a previous patch
  - Combine patches
  - Reorder patches
- We can do this on our local clone (typically to clean up some detours in development)
- You can never do this once you've published code to a public repository

MATH 676

–

Finite element methods in scientific computing

Wolfgang Bangerth, Texas A&M University