The version control system git

Norbert Manthey
International Center for Computational Logic
Technische Universität Dresden
Germany

- Why do we need a version control system?
- How to use it - step by step
- What makes git different?

"Logic is everywhere ..."
What is the problem?

- Assume: Peter and I write a paper
- Assume: Peter writes section 1,3 and 5, I write section 2 and 4

Peter

Norbert

- sets up the document
What is the problem?

- Assume: Peter and I write a paper
- Assume: Peter writes section 1, 3 and 5, I write section 2 and 4

  Peter
  - sets up the document
  - sends it to me

  Norbert
  - waits for section 2
  - writes section 3, and sends

Peter Manthey
The version control system git
What is the problem?

► Assume: Peter and I write a paper

► Assume: Peter writes section 1, 3 and 5, I write section 2 and 4

Peter

► sets up the document

► sents it to me

► waites for section 2

Norbert

This time: just copy a whole section
What is the problem?

- Assume: Peter and I write a paper
- Assume: Peter writes section 1, 3 and 5, I write section 2 and 4

<table>
<thead>
<tr>
<th>Peter</th>
<th>Norbert</th>
</tr>
</thead>
<tbody>
<tr>
<td>sets up the document</td>
<td></td>
</tr>
<tr>
<td>sents it to me</td>
<td></td>
</tr>
<tr>
<td>waites for section 2</td>
<td></td>
</tr>
<tr>
<td></td>
<td>write section 2, start section 4</td>
</tr>
</tbody>
</table>
What is the problem?

- Assume: Peter and I write a paper
- Assume: Peter writes section 1, 3 and 5, I write section 2 and 4

Peter
- sets up the document
- sends it to me
- waits for section 2
- writes section 3, and sends

Norbert
- write section 2, start section 4
What is the problem?

- Assume: Peter and I write a paper
- Assume: Peter writes section 1, 3 and 5, I write section 2 and 4

Peter
- sets up the document
- sends it to me
- waits for section 2
- writes section 3, and sends

Norbert
- writes section 2, start section 4
- receive file, merge
What is the problem?

- Assume: Peter and I write a paper
- Assume: Peter writes section 1, 3, and 5, I write section 2 and 4

**Peter**
- sets up the document
- sends it to me
- waits for section 2

**Norbert**
- writes section 3, and sends
- write section 2, start section 4
- receive file, **merge**

- **This time**: just copy a whole section
Is this it?

- Copying a whole passage is simple, however:
  - Who told you which part has been changed?
  - What should Peter do if Peter finds a typo in my text?
  - What should I do while I am waiting for Peter to finish his section?
Is this it?

- Copying a whole passage is simple, however:
  - Who told you which part has been changed?
  - What should Peter do if Peter finds a typo in my text?
  - What should I do while I am waiting for Peter to finish his section?
- There exists **diff** tools . . .
Is this it?

- Copying a whole passage is simple, however:
  - Who told you which part has been changed?
  - What should Peter do if Peter finds a typo in my text?
  - What should I do while I am waiting for Peter to finish his section?
- There exists diff tools . . .

- Can we work at the same document concurrently?
- Can I work offline?
- How does it work?
How does it work?

- Assume: Peter and I write a paper with the version control system *git*
- Assume: Peter and I both have a local copy

Peter
- sets up the document, and *git*

Norbert
How does it work?

- Assume: Peter and I write a paper with the version control system **git**
- Assume: Peter and I both have a local copy

**Peter**

- sets up the document, and **git**
- sents the repository

**Norbert**

- clones repository, writes text
- commit changes locally
- pull changes (global repository)
- merges automatically
- push changes (global repository)
How does it work?

- Assume: Peter and I write a paper with the version control system git
- Assume: Peter and I both have a local copy

<table>
<thead>
<tr>
<th>Peter</th>
<th>Norbert</th>
</tr>
</thead>
<tbody>
<tr>
<td>sets up the document, and git</td>
<td></td>
</tr>
<tr>
<td>sents the repository</td>
<td></td>
</tr>
<tr>
<td>starts section 1, adds references</td>
<td></td>
</tr>
</tbody>
</table>
How does it work?

▶ Assume: Peter and I write a paper with the version control system git
▶ Assume: Peter and I both have a local copy

Peter
▶ sets up the document, and git
▶ sends the repository
▶ starts section 1, adds references
▶ commit changes locally

Norbert
▶ clones repository, writes text
How does it work?

- Assume: Peter and I write a paper with the version control system `git`
- Assume: Peter and I both have a local copy

Peter
- sets up the document, and `git`
- sends the repository
- starts section 1, adds references
- commit changes locally
- push changes to global repository

Norbert
- clones repository, writes text
- commit changes locally
How does it work?

- Assume: Peter and I write a paper with the version control system git
- Assume: Peter and I both have a local copy

- Peter:
  - sets up the document, and git
  - sents the repository
  - starts section 1, adds references
  - commit changes locally
  - push changes to global repository

- Norbert:
  - clones repository, writes text
  - commit changes locally
  - pull changes (global repository)
How does it work?

- Assume: Peter and I write a paper with the version control system git
- Assume: Peter and I both have a local copy

**Peter**
- sets up the document, and git
- sends the repository
- starts section 1, adds references
- commit changes locally
- push changes to global repository
- continues writing

**Norbert**
- clones repository, writes text
- commit changes locally
- pull changes (global repository)
- merges automatically
How does it work?

► Assume: Peter and I write a paper with the version control system git

► Assume: Peter and I both have a local copy

<table>
<thead>
<tr>
<th>Peter</th>
<th>Norbert</th>
</tr>
</thead>
<tbody>
<tr>
<td>► sets up the document, and git</td>
<td>► clones repository, writes text</td>
</tr>
<tr>
<td>► sents the repository</td>
<td>► commit changes locally</td>
</tr>
<tr>
<td>► starts section 1, adds references</td>
<td>► pull changes (global repository)</td>
</tr>
<tr>
<td>► commit changes locally</td>
<td>► merges automatically</td>
</tr>
<tr>
<td>► push changes to global repository</td>
<td>► push changes (global repository)</td>
</tr>
<tr>
<td>► continues writing</td>
<td>► continues writing</td>
</tr>
<tr>
<td>► continues writing</td>
<td></td>
</tr>
</tbody>
</table>
How does it work?

► Requirements:
  ▶ A server where the global repository is stored
  ▶ Access to this server for each participant

► What are the basic steps?
  ▶ Set up the git repository
  ▶ Create a local copy
  ▶ Add files to the repository
  ▶ Commit local changes
  ▶ Pull changes from the global repository
  ▶ Push changes to the global repository
  ▶ In a conflict case: merge manually

► You can do much more, e.g. branch, roll back, ...
How to set up a git repository?

▶ Choose a server and create a directory
▶ Create a repository there

```bash
ssh peter@spock
mkdir git
cd git
git init --bare --shared=group .
```

▶ `git init` creates the repository
▶ `--bare` creates the repository as not-useable
▶ `--shared=group` the repository can be shared with others from the same UNIX group
▶ `.` create the repository here
How to create a local copy?

▶ **Clone the global repository to your hard disk**

```bash
mkdir paper
cd paper
git clone norbert@spock:~peter/git .
```

▶ **git clone** clones a repository to the specified place

▶ **spock: peter/git** is the link to the repository on the server

▶ **norbert** is my own user name on that server

▶ **.** create the repository here
How to add files to the repository?

▶ Create files and adding them to the repository

```
echo "hello" >> paper.tex
git add paper.tex
```

▶ `git add` adds files to the current **local** repository

▶ from now on, any following `git` command cares about this file
How to add files to the repository?

► Create files and adding them to the repository

```bash
echo "hello" >> paper.tex
git add paper.tex
git commit -am "added first file"
```

► `git add` adds files to the current local repository

► from now on, any following `git` command cares about this file

► `git commit` records the changes locally

► `-am "added first file"` tells `git` to record all modifications and all added files

► in the same way, file modifications can be recorded by the repository
How to pull changes from the global repository?
► Recording everything locally does not share the data
► The global repository might contain new data from others
► To get their data, changes need to be pulled

```
git pull origin master
```

► `git pull` download changes from the global repository
► `origin master` tells `git` where to get the changes from
► `origin` is the cloned repository
► `master` is the main branch
How to pull changes from the global repository?

- Recording everything locally does not share the data
- The global repository might contain new data from others
- To get their data, changes need to be **pulled**

```bash
git pull origin master
```

```bash
git commit -am "added first file"
```

- **git pull** download changes from the global repository
- **origin master** tells **git** where to get the changes from
- **origin** is the cloned repository
- **master** is the main branch
- Pulling changes might result in **conflicts**
- **Conflict files** need to be merged **manually**
- In each conflict file, conflicts are surrounded by "<<<" and ">>>"
- After manual merging, the conflict file needs to be **added** and **committed** again
How to push changes to the global repository?

► Recording everything locally does not share the data
► To share your data, you need to push it into the global repository

```
git push origin master
```

► `git pull` download changes from the global repository
► `origin master` is the same as for `git pull`
► Note: good practice is to always pull before pushing
Some limits

- How does git work?
  - Basically, it is comparing altered lines
- What happens if both parties change the same line?
  - git will show a conflict
- What happens if both parties change the empty line below line $X$?
  - git will show a conflict
- One way to tackle this for LaTeX?
  - Brake lines frequently
- Still, the automatic merging is very powerful!
Working on your thesis

- You can also use the local repository only
  - Set up the repository (you want to use it and not share it)
    git init .
  - You can use this repository as backup/history
  - As usual, you can add files
    git add
  - You can commit changes
    git commit -am "comment"
- Of course, you could also use a global repository alone
To sum up

► In the extreme case, first one user . . .
  ▶ sets up the repository `git init --bare --shared=group`.
  ▶ clones the repository `git clone me@server:~me/pathToGit`.
  ▶ adds all files `git add` and `git commit -am "comment"`
  ▶ prepares the global repository `git push origin master`
  ▶ and shares the link to the repository, e.g. `server:~me/pathToGit`

. . . and all users do

  ▶ Clone the repository `git clone you@server:~me/pathToGit`.
  ▶ Change files
  ▶ Record changes locally `git commit -am "comment"`
  ▶ Get the latest global changes `git pull origin master` (and sometime merge)
  ▶ Share local changes `git push origin master`
  
  and repeat the last four steps until the document is final
Conclusion

- Working on a shared document concurrently is easy
- Why should you use **git** and not **svn** or **cvs**?
  - You have a local repository
  - You can jump back to old versions locally (also in the plane)
- Is there a nice GUI?
  - Linux: **gitg** or **gitk**
  - Mac: **sourcetree** or **gitx**
- Still, it's only four command line commands to know …