Using a computer as a computer scientist

Louis Jachiet

Digital life

Digital life

Mails – some general and technical considerations

What is a mail?

From a technical standpoint a mail is just a text file looking like this:

```
From: Louis Jachiet <test@jachiet.com>
To: Louis Jachiet <louis.jachiet@telecom-paris.fr>
Cc: cc-ed@jachiet.com
Subject: Re: This is a subject
Message-ID: <20220903164909.uksb6uca3o2zyyd4@moa.enst.fr>
MIME-Version: 1.0
Content-Type: text/plain; charset=us-ascii
Content-Disposition: inline
Content-Transfer-Encoding: quoted-printable
In-Reply-To: <20220903164842.fscuyipdyhc67pxk@moa.enst.fr>

Well received!
--
Louis Jachiet
```

What is a mail?

To send a mail, you contact a (mail) server that will take this file and send it to another server, that will send it to another server until it reaches the inbox of the recipients.

Advantages

Mail is somewhat standard

sending text and files to almost anyone

Mail is federated

does not belong to a single entity

Mail is asynchronous

generally instantaneous

Disadvantages

Many extensions are non standard

encryption, receipt, colorful mails, etc.

Mail is federated

each provider runs its own instracture which can create friction

Mail is asynchronous

mail is not always fast...

How not to write an email

From: duck42@spacelasers.com
To: Prof 3
Cc: Prof 2, Random person
Bcc: Prof 1
Subject: Mail

I had trouble with computers last week first I tried to reinstall my computer and it failed then I decided to use my friend computer but I had to start over and I was no able to send my work yesterday. Here is my file

Attachment: tp.rar
What is wrong?

Use a proper subject!

Keep it relatively short, but informative.

Be polite!

Unless you are replying to a recent mail:

- Add a salutation (Hello, Hi, Dear Teacher, Dear Ms X, etc.)
- Add an appropriate closing (Sincerely, Regards, Best Regards, Thank you)
- Use the correct language level
- Never write an email when you are angry / always down tone your angryness
- Don't use sarcasm

Be specific!

We you received hundreds of mails a day you spend 30s reading them (sometimes doing something else in parallel).

Each mail should be very specific in what you want!

Give context!

We often assume symmetric relation and that readers are our mails will have the same mental context as ours. *It is not the case!*

It is always better to add too much context!

Too much information

Focus on what is *essential*.

Some advice

- The shape of your mail helps reading it!
- We (generally) don't need to know why you were late, sick, etc.
- Often, the body of a mail can fit the 280 chars tweet length

Timely follow up

We all loose tracks of mails sometimes, so you can send a reminder but:

- we have other things to do
- we might take week-ends and vacation
- the urgency for you is not a criterion if you are the reason for that urgency

---

Mails should ideally be sent during a week day before 5 pm.

Proof-read your mail

- Re-read everything you wrote
- Check all items above
- Check if attachments are needed

Digital life

How to use mail for Data AI

How to write an email

- Use a proper mail address
- Select carefully the recipients
- Use a proper subject
- Be polite
- Be brief but describe:
  - the problem
  - your attempt
  - an eventual (short but VALID) excuse
- For attachments :
  - use meaningful names
  - use standardized file types (e.g. PDF, ZIP) not docx or pages
  - remove useless files
  - keep size small (if possible)

How to write an email

From: first.lastname@telecom-paris.fr
To: Main teacher
Cc: TAs
Subject: [DATAAI 101] Late submission for assignment XY

Dear teacher,

I am firstname LASTNAME and I am in your course DATAAI 101. We were supposed to submit the TP 4 on moodle before the 5th but sadly I missed the deadline and now the submission on moodle is closed.

I am really sorry for this but would you consider the attached file? Or allow me to submit on moodle?

Warm Regards,
–
Firstname Lastname
Attachment: LASTNAME_firstname_TP4_DATAAI_101.zip

Using the Télécom mail

Your `@telecom-paris.fr` address is not forever!

It will disappear a few months after your departure,

be wary when communicating with this address!

Important communications will be sent to this address,

check it regularly!

It comes with a calendar tool, don't hesitate to use such a tool!

Digital life

Security

Threat model

For each data storage determine:

- Who has access to it
- Against what I want to protect myself?
- How do I protect myself against it?

Résumé

Confidentiality

Very low

Could be put on my webpage?

Résumé

Confidentiality

Very low

Could be put on my webpage?

- If it includes your mail, what about spam?

Résumé

Confidentiality

Very low

Could be put on my webpage?

- If it includes your mail, what about spam?
- If if includes a lot of personal infos how to protect “password lost” questions and identity theft?

Bank Account Number / IDs

Confidentiality

Low

Who can use it?

Bank Account Number / IDs

Confidentiality

Low

Who can use it?

- Scammers.

Personal mail

Confidentiality

High

What are the risks?

Personal mail

Confidentiality

High

What are the risks?

- Scammers.

Personal mail

Confidentiality

High

What are the risks?

- Scammers.
- Extortion.

Personal mail

Confidentiality

High

What are the risks?

- Scammers.
- Extortion.
- Intimate mails being leaked

Personal mail

Confidentiality

High

What are the risks?

- Scammers.
- Extortion.
- Intimate mails being leaked
- Stolen online accounts

Personal mail

Confidentiality

High

What are the risks?

- Scammers.
- Extortion.
- Intimate mails being leaked
- Stolen online accounts
- ...

How to secure your data?

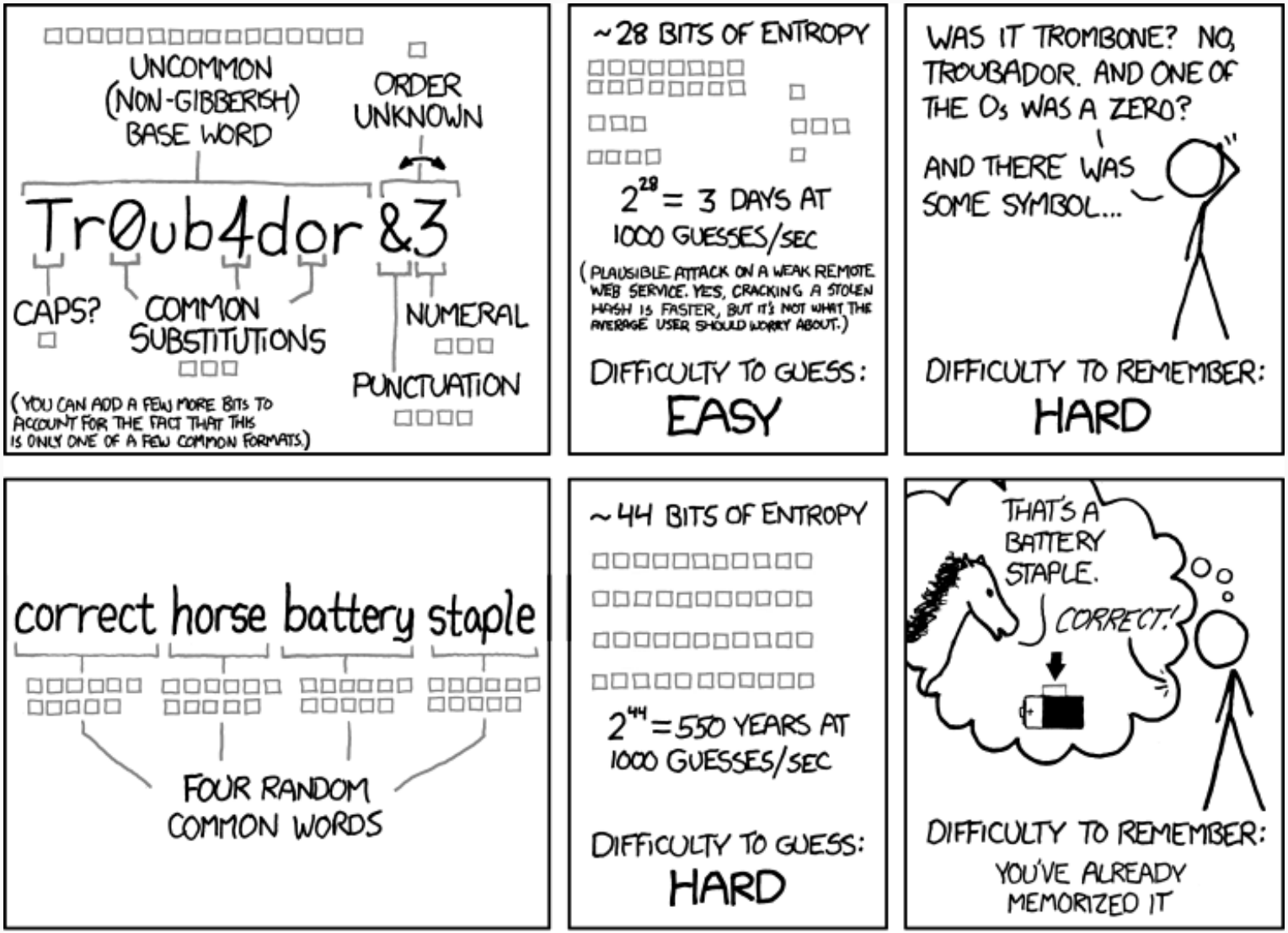
Passwords

- A different password for each website (use a password manager!)
- Strong passwords

Storage devices

- Encrypt all the drives containing data
- Don't use random USB keys

Crafting passwords?



Crafting passwords

Algorithm

- Select 3 to 4 random words
- Separate them with random numbers or punctuation

Example

- mechanization / preference / outback / apologizing
- Separate them with numbers or punctuation

Result

mechanization4preference;outback_apologizing

Entropy

$30\,000$ words $\rightarrow 30\,000^4 \approx 10^{18}$ possibilities

Digital life

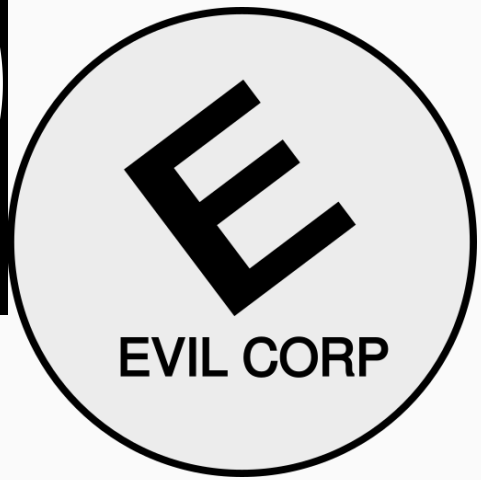
Backups

Who is the biggest enemy of your data?

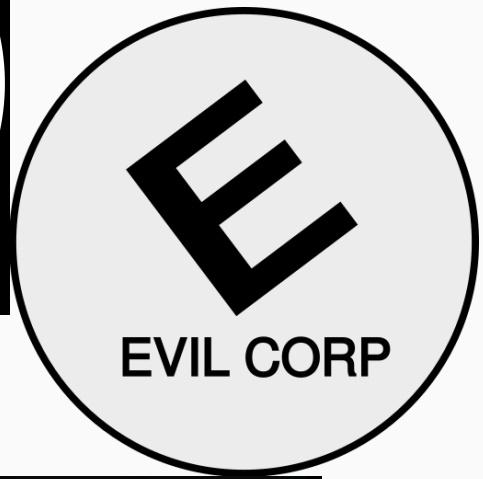
Who is the biggest enemy of your data?

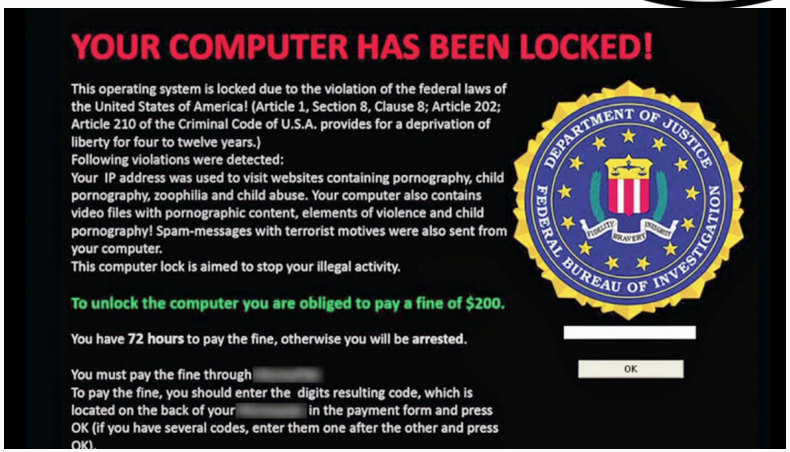
Who is the biggest enemy of your data?

Who is the biggest enemy of your data?



Who is the biggest enemy of your data?





Who is the biggest enemy of your data?

What can happen to your data?

- Loose it
- Being stolen
- Have an accident (fire, flood, etc.)
- Hard disk failure
- Data corruption
- Accidental deletion
- ...

One of these items happens every few years to everyone...

How to backup data

Golden rule: 3 times

- three different storage types,
- three different places,
- three types of backups.

Easy mistake: can you recover?

It is important to check that EVERYTHING is really backed-up from time to time and that the data is really there:

- Closed backup provider
- Corrupted USB key
- Some crucial files are missing
- etc.

Digital life

Online presence

What is online will stay online...

but you can somewhat control it by adding content.

Creating a web page

Creating a webpage can be done in a free and easy way, for instance :

- Github pages
- Gitlab
- Wordpress
- Netlify
- and many other possibilities...

Tools for computer scientists

Tools for computer scientists

Bibliography

Bibliography

Producing science requires to be up to date on recent advances.

Bibliography

Producing science requires to be up to date on recent advances.
In computer science, science is first published in research papers (theses, journals, conferences, etc.).

Bibliography

Producing science requires to be up to date on recent advances. In computer science, science is first published in research papers (theses, journals, conferences, etc.).

To pursue science you need to find, get access, read, remember and cite the relevant article for your research.

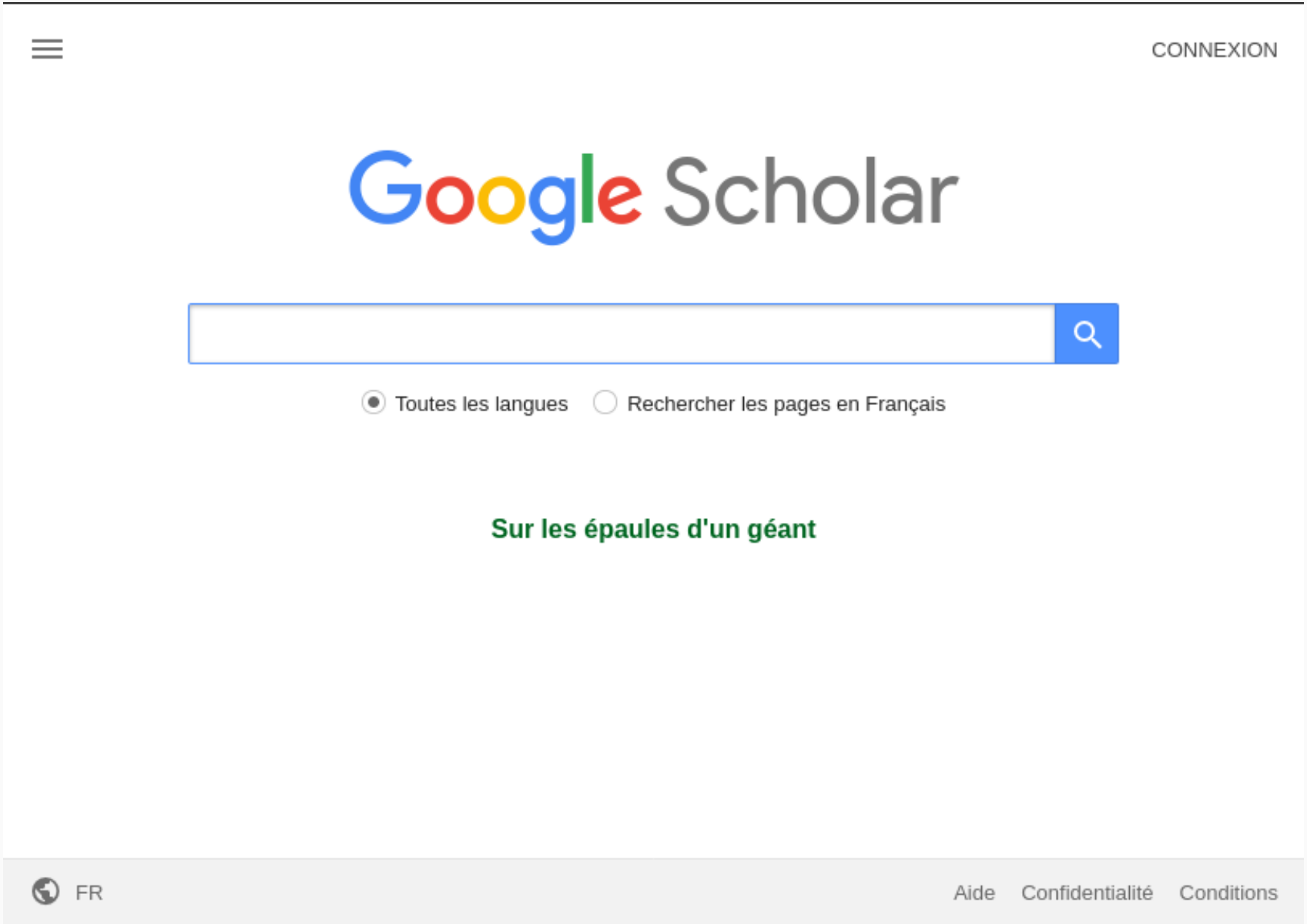
Finding articles

Research articles are generally not easy to read. When starting a completely new subject, it is a good idea to read blog posts or books about it rather.

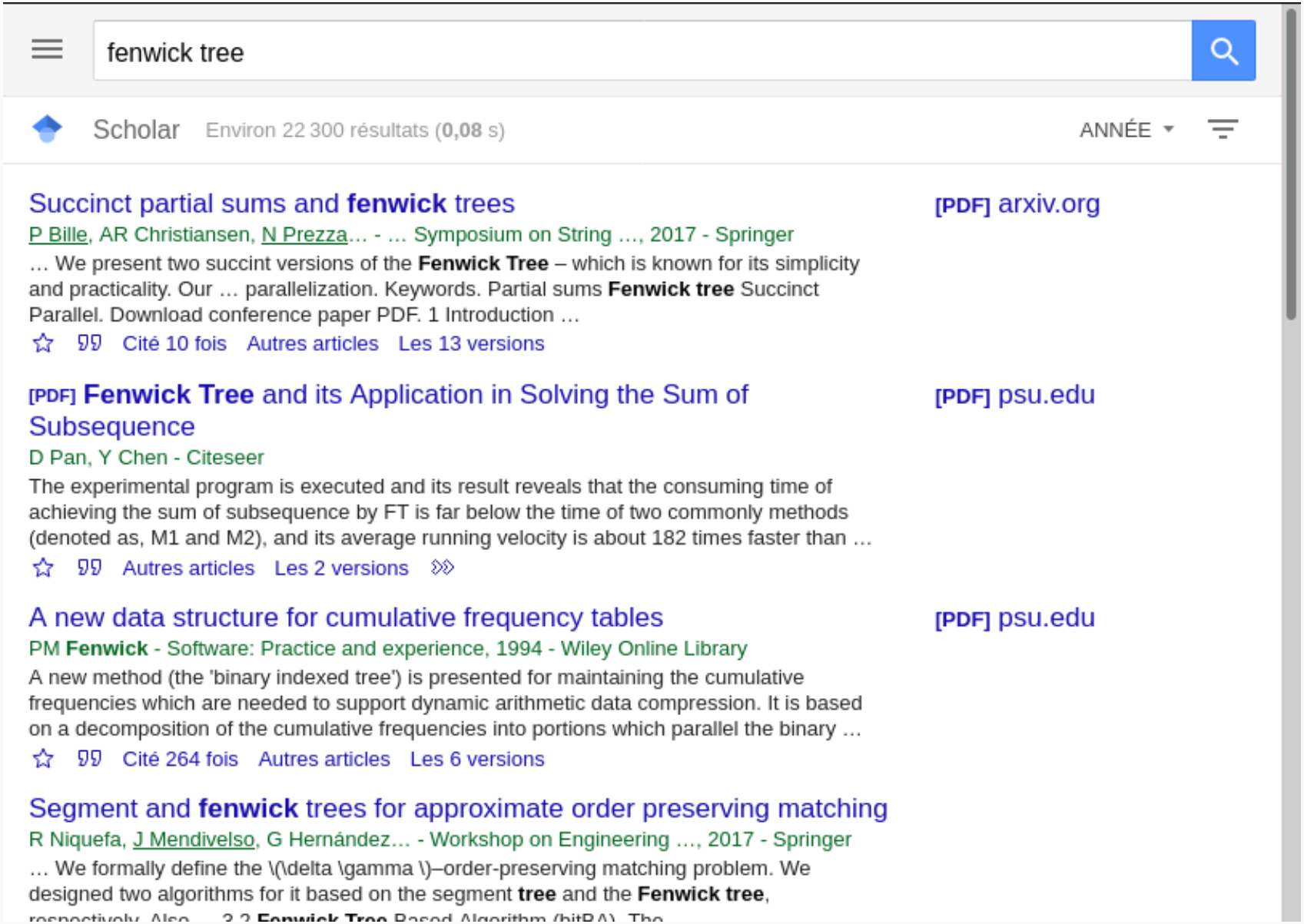
When starting an internship or a PhD most of the bibliography will be provided by your advisor.

When looking for a specific reference or with a specific question you can use a search engine, the most popular being Google Scholar.

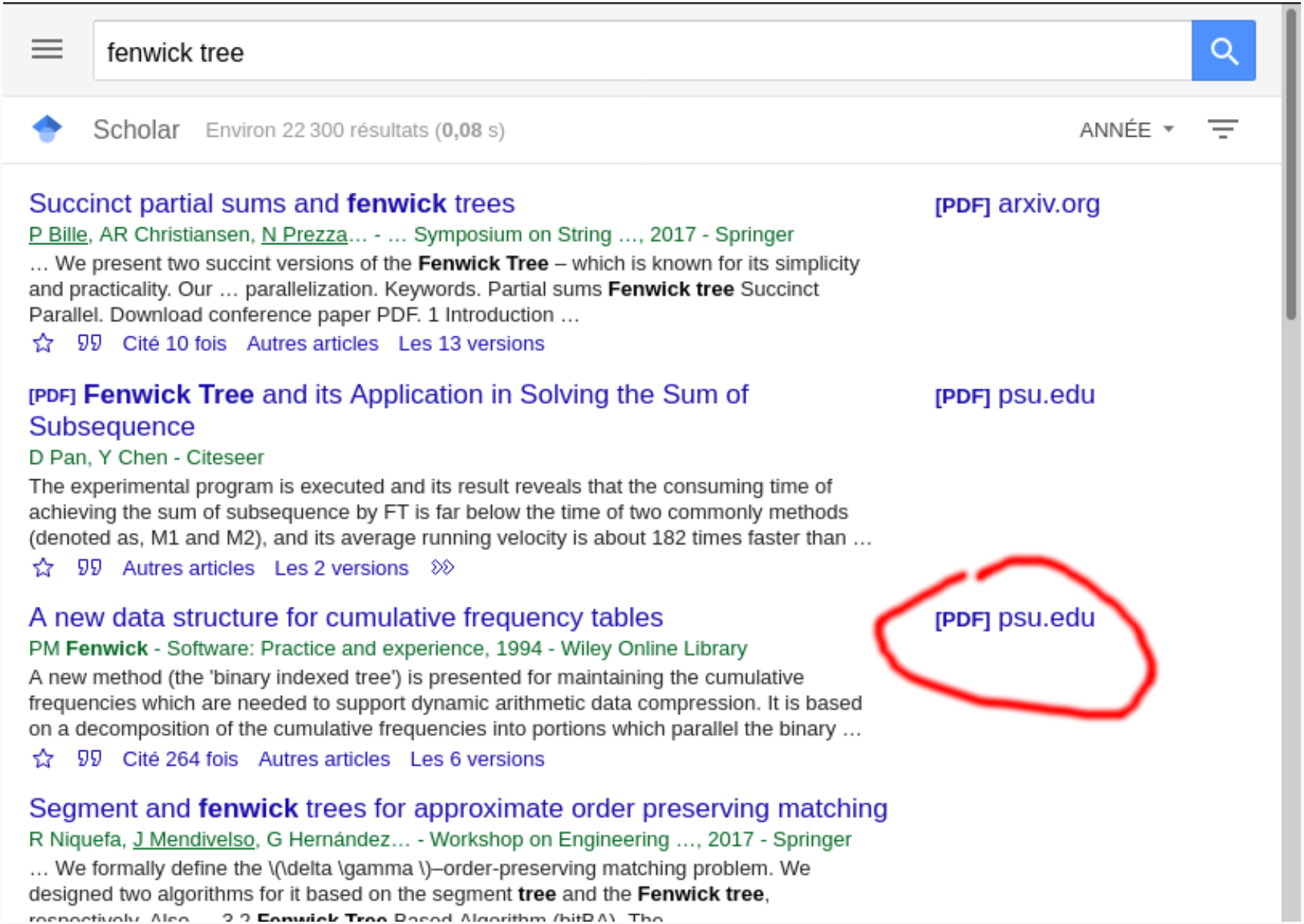
Finding article with Google scholar



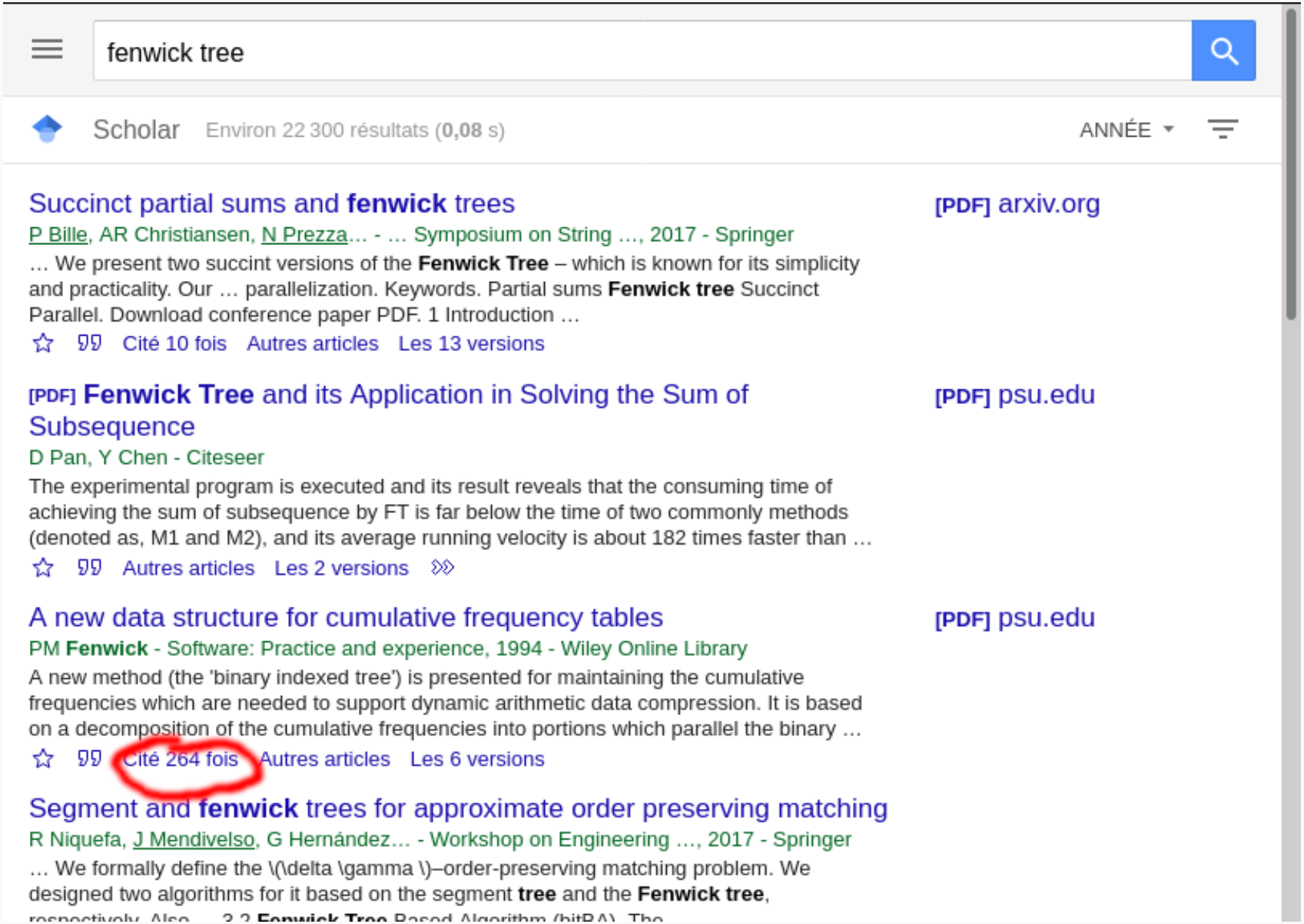
Finding article with Google scholar



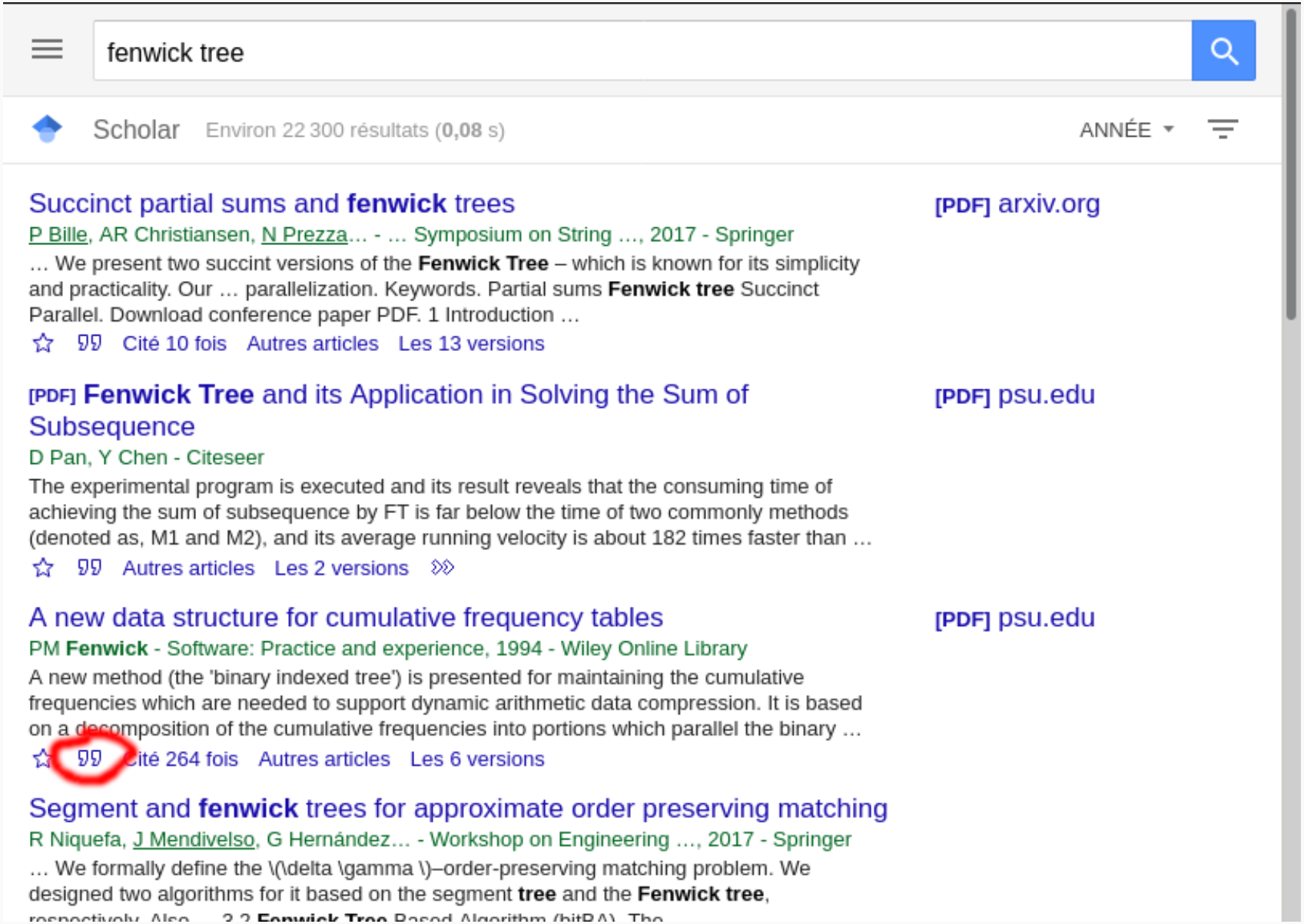
Finding article with Google scholar



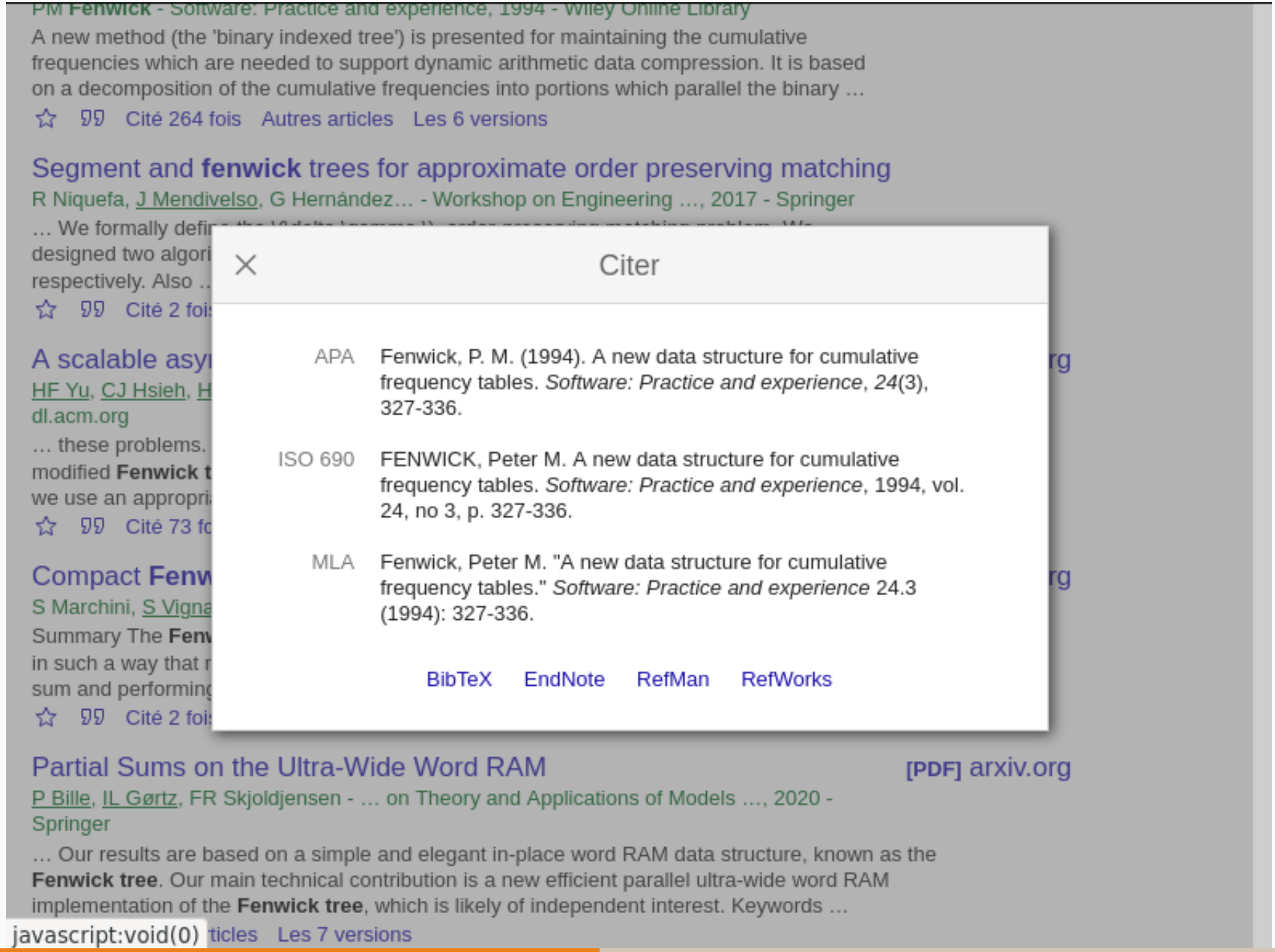
Finding article with Google scholar



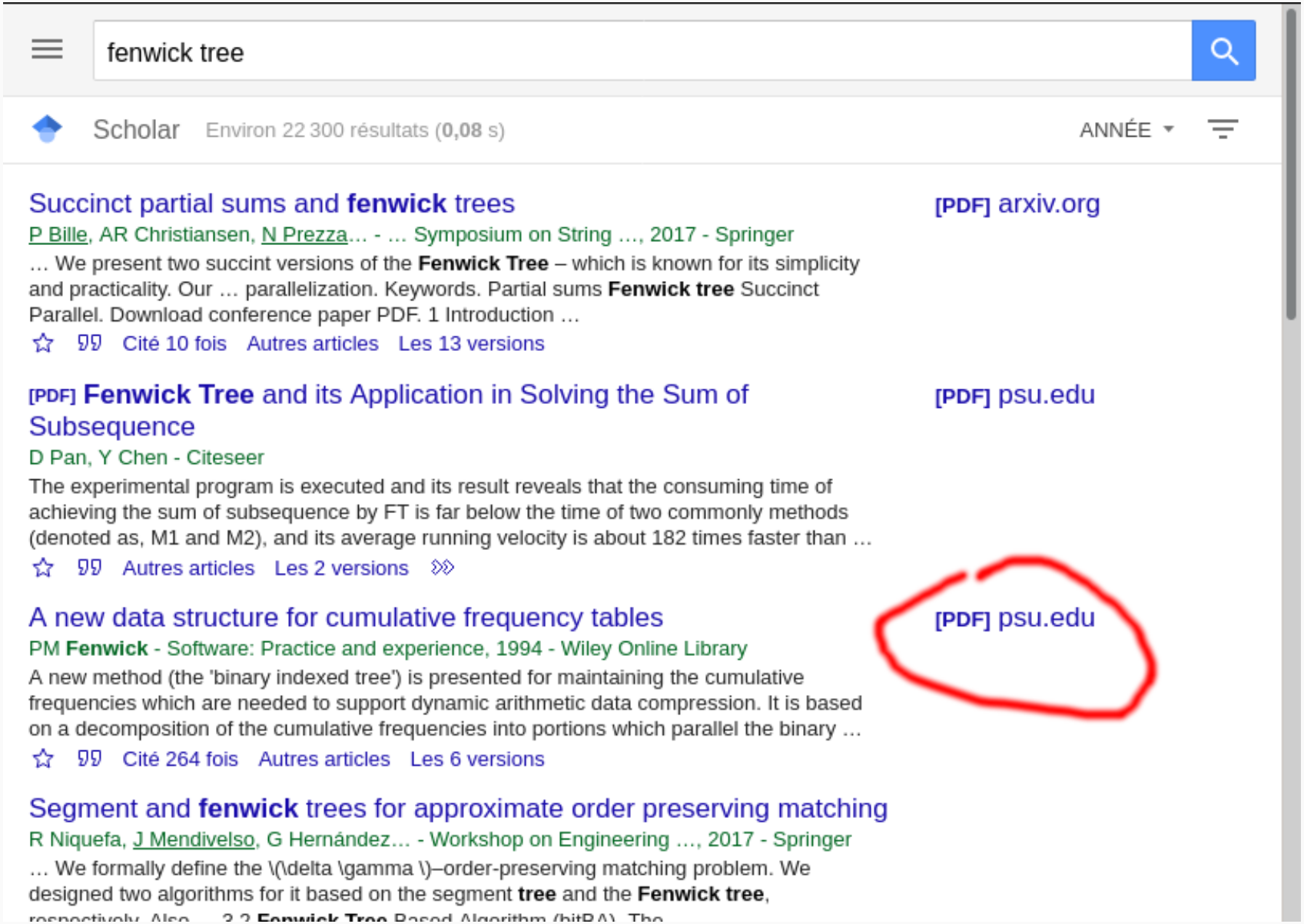
Finding article with Google scholar



Finding article with Google scholar



Getting access to articles



Getting access to articles

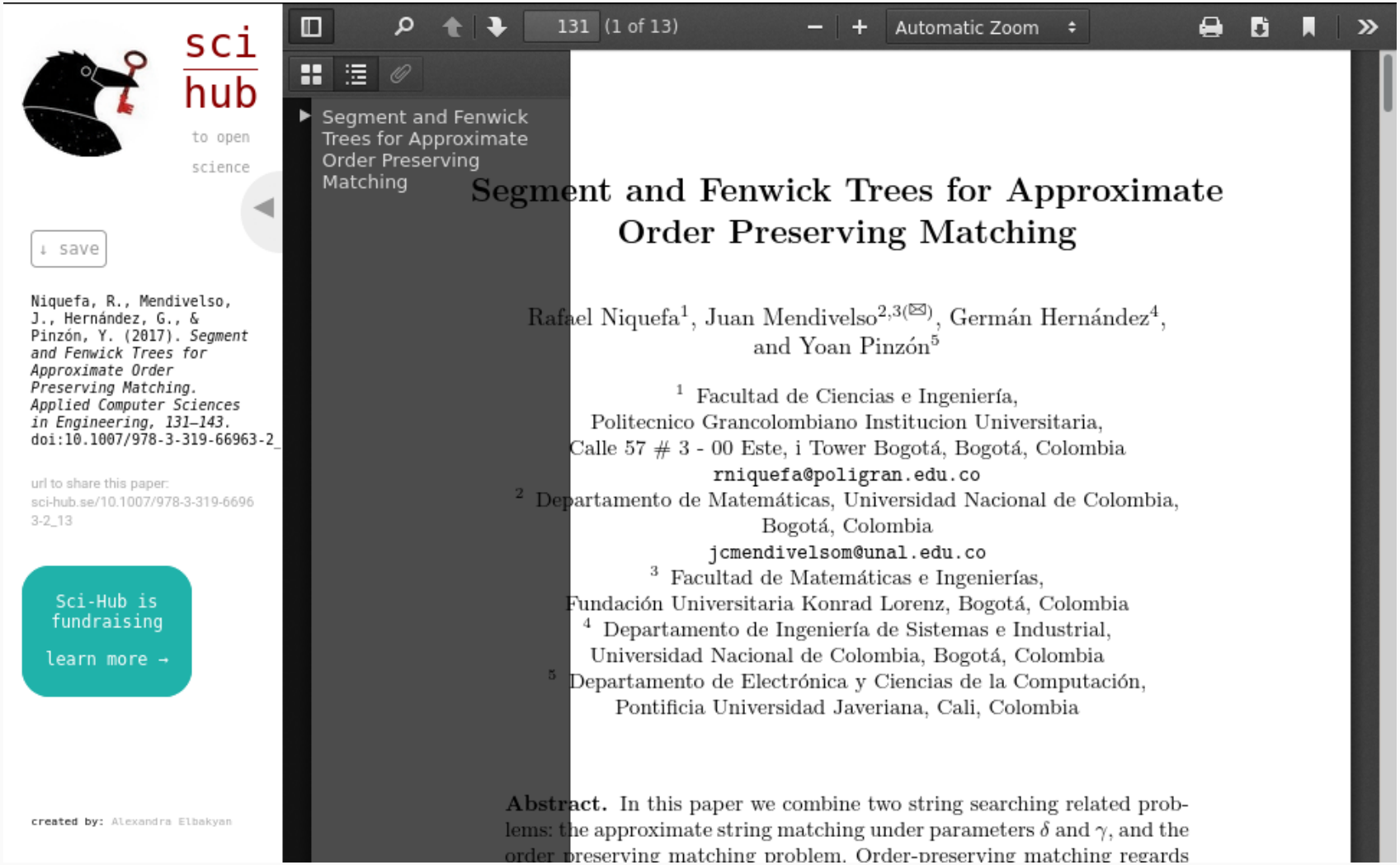
Some articles are behind a paywall what can you do:

- See if Google scholar found an opened version
- Access through Télécom (Télécom is subscribed to several publishers)
- Use libgen / sci-hub (legality to discuss)

Getting access to articles with sci-hub

Getting access to articles

Getting access to articles



Read and remember articles

When you have read 3 articles in the past month, it is easy to remember:

- What each article says
- In what version of the article you can find the result
- The precise reference of the article

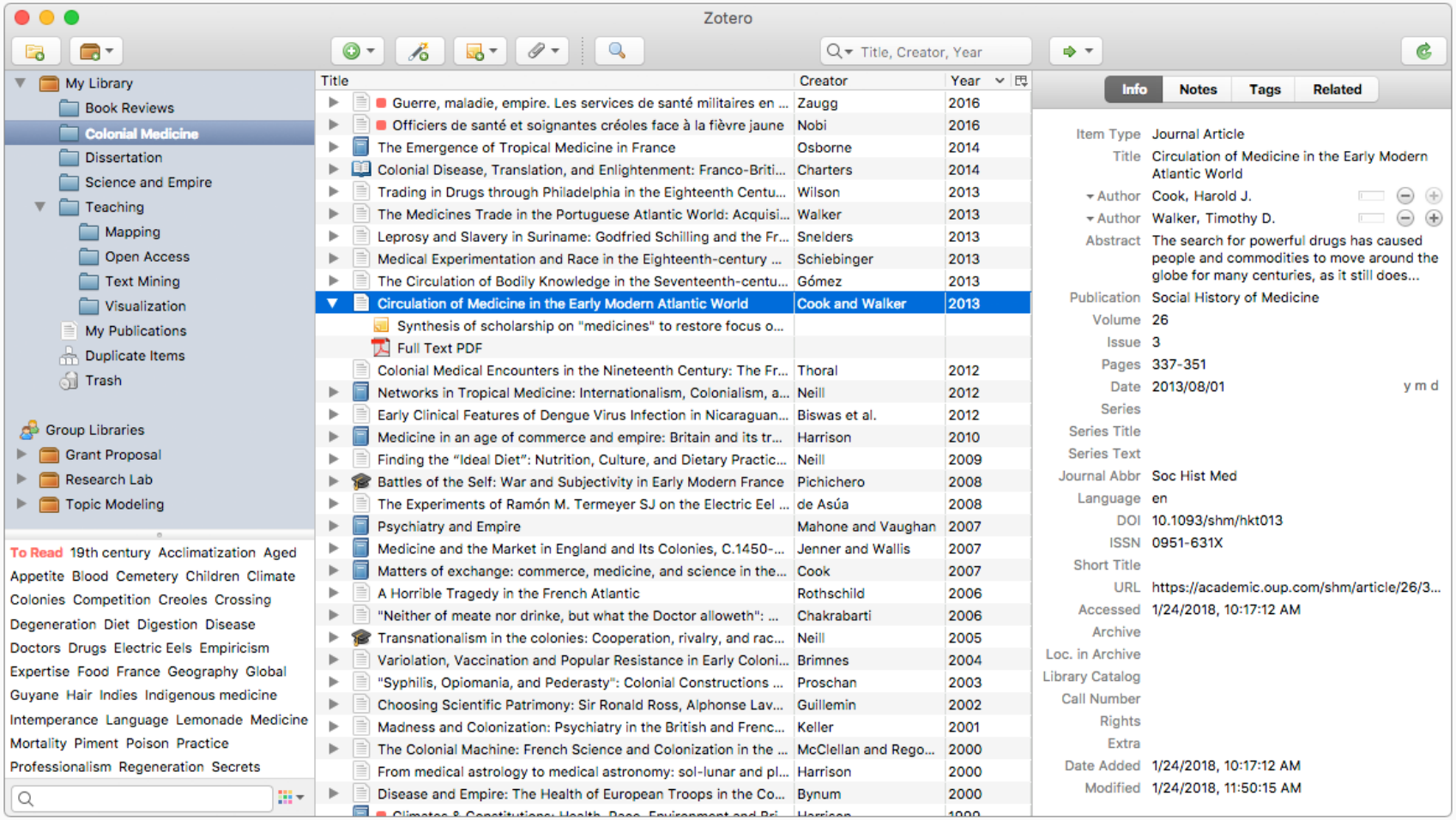
After three years (duration of PhD in France) and 100s of articles, it is much harder...

Read and remember articles

It is a very good thing to have *some system* to keep track of which article you read and what it contained as well as the BibTeX reference.

- A folder with the PDF of articles and notes
- A wiki or some other note taking tool
- Zotero

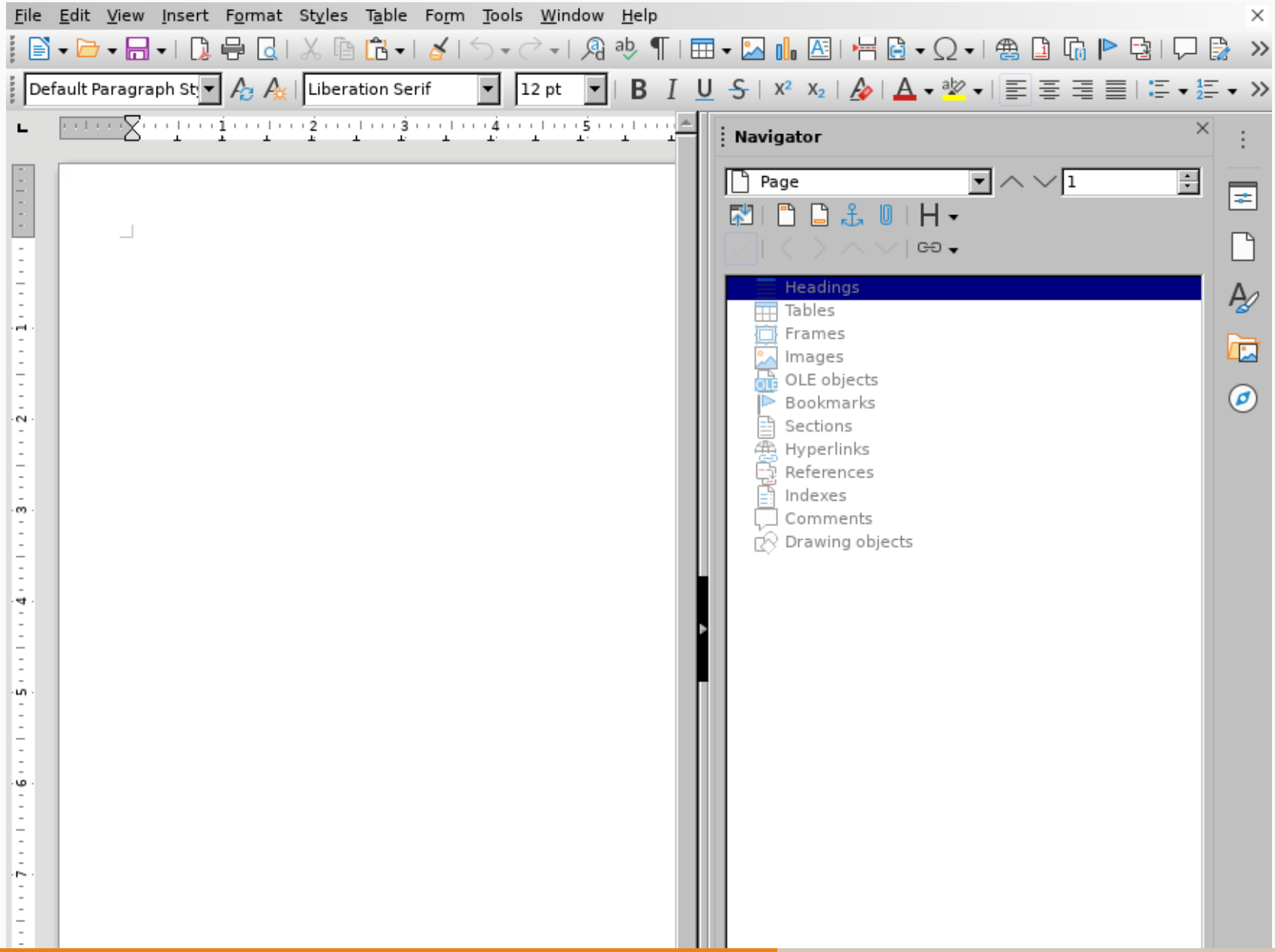
Read and remember articles with Zotero



Tools for computer scientists

Writing research

Office


File Edit View Insert Format Styles Table Form Tools Window Help
Default Paragraph St
Liberation Serif
12 pt
Navigator
Page
1
Headings
Tables
Frames
Images
OLE objects
Bookmarks
Sections
Hyperlinks
References
Indexes
Comments
Drawing objects

In CS, most conferences expect articles to be formatted through LaTeX which is a software layer over TeX.

LaTeXcode for this slide

```
\begin{frame}{\LaTeX}
  In CS, most conferences expect articles to be formatted
  through {\LaTeX} which is a software layer over {\TeX}.

  \hle

  \inputmintedbox{LaTeX}{recslide.tex}
                 {\LaTeX code for this slide}
\end{frame}
```

Pros

- Easy to write science
- Separation of styling and content
- Can be “programmed”
- Works well with git and so
- Good looking documents by default

Cons

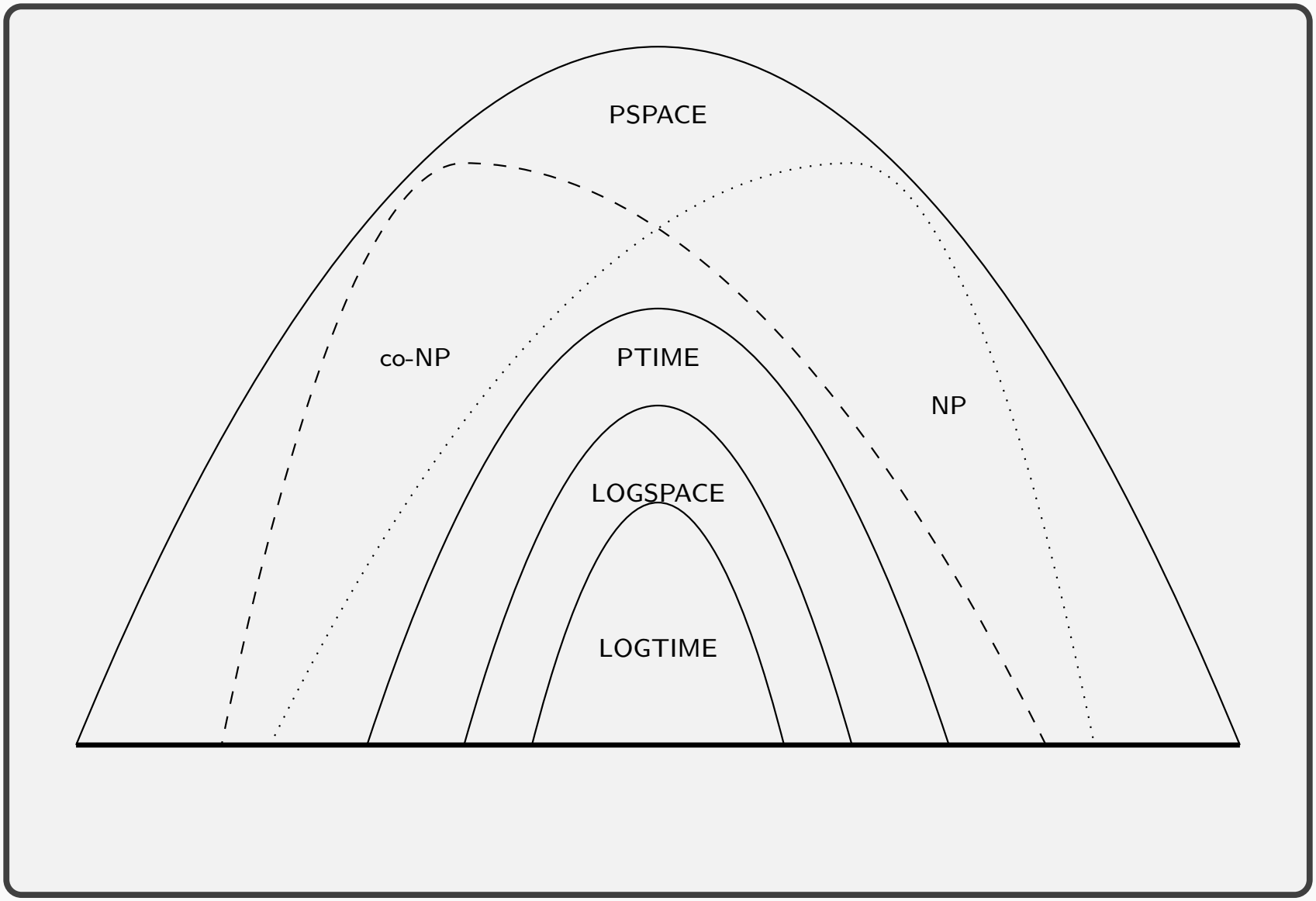
- Learning curve is hard for first timers
- Requires help to do something new
- Requires installation of software to write

One big advantage of LaTeXis that there exists templates for a multitude of use cases:

- research articles
- presentations
- résumé
- letters
- ...

Drawing figures with Tikz

Research articles often contains complex figures such as:



Drawing figures with Tikz

Research articles often contains complex figures such as:

```
\begin{tikzpicture}
  \tiny \pgftransformscale{.8}
  \draw[very thick] (6,0) -- (-6,0);
  \draw (-1.3,0) parabola bend (0,2.5) (1.3,0) ; \node at (0,1) { LOGTIME };
  \draw (-2,0) parabola bend (0,3.5) (2,0); \node at (0,2.6) { LOGSPACE };
  \draw (-3,0) parabola bend (0,4.5) (3,0); \node at (0,4) {PTIME};
  \draw[dotted] (-4,0) parabola bend (2,6) (4.5,0);  \node at (3,3.5) {NP};
  \draw[dashed] (4,0) parabola bend (-2,6) (-4.5,0); \node at (-2.5,4) {co-NP};
  \draw (-6,0) parabola bend (0,7.2) (6,0); \node at (0,6.5) {PSPACE};
\end{tikzpicture}
\vspace{-2em}
```

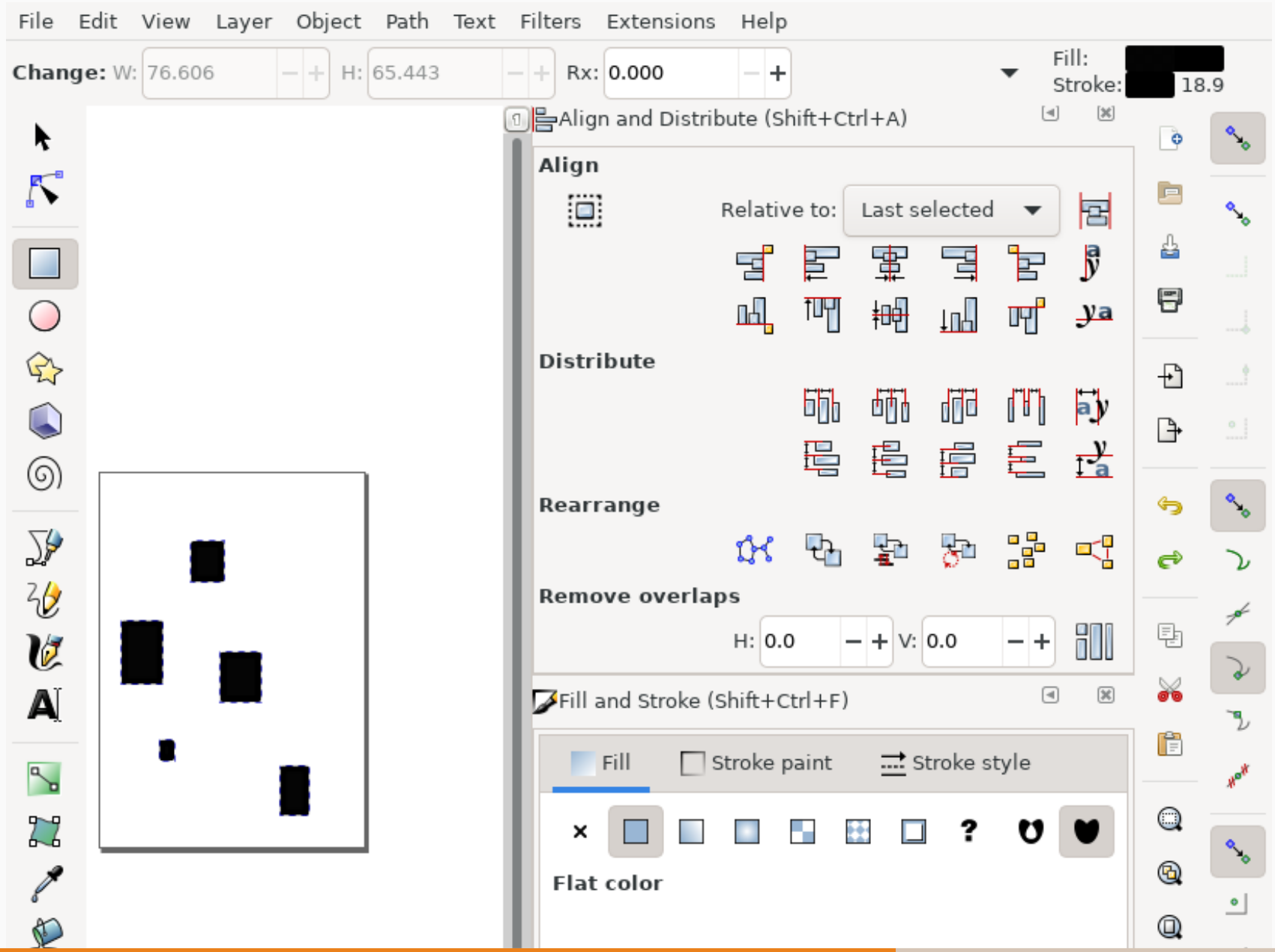
Figure 1: Example of Tikz use from texample.net

Drawing figures with inkscape

For articles, always prefer vector graphics to raster graphic:

- lighter
- scalable
- editable

Drawing figures with inkscape



Tools for computer scientists

Version control systems

Version Control Systems (VCS)

A system eventually capable of:

- Capture the evolution of files
- Synchronize files across multiple computers
- Allow multiple users to work in parallel on the same documents

Version Control Systems (VCS)

A system eventually capable of:

- Capture the evolution of files
- Synchronize files across multiple computers
- Allow multiple users to work in parallel on the same documents

It can be applied to a wide range of types of files:

- Source code
- Websites
- Reports
- Work environments
- ...

Popular VCS

The most popular ones are:

- Git
- Bazaar
- mercurial
- SVN (outdated)
- RCS (very outdated)

Around git

While Git works in a fully decentralized way but whole ecosystems have emerged around it with version control but also:

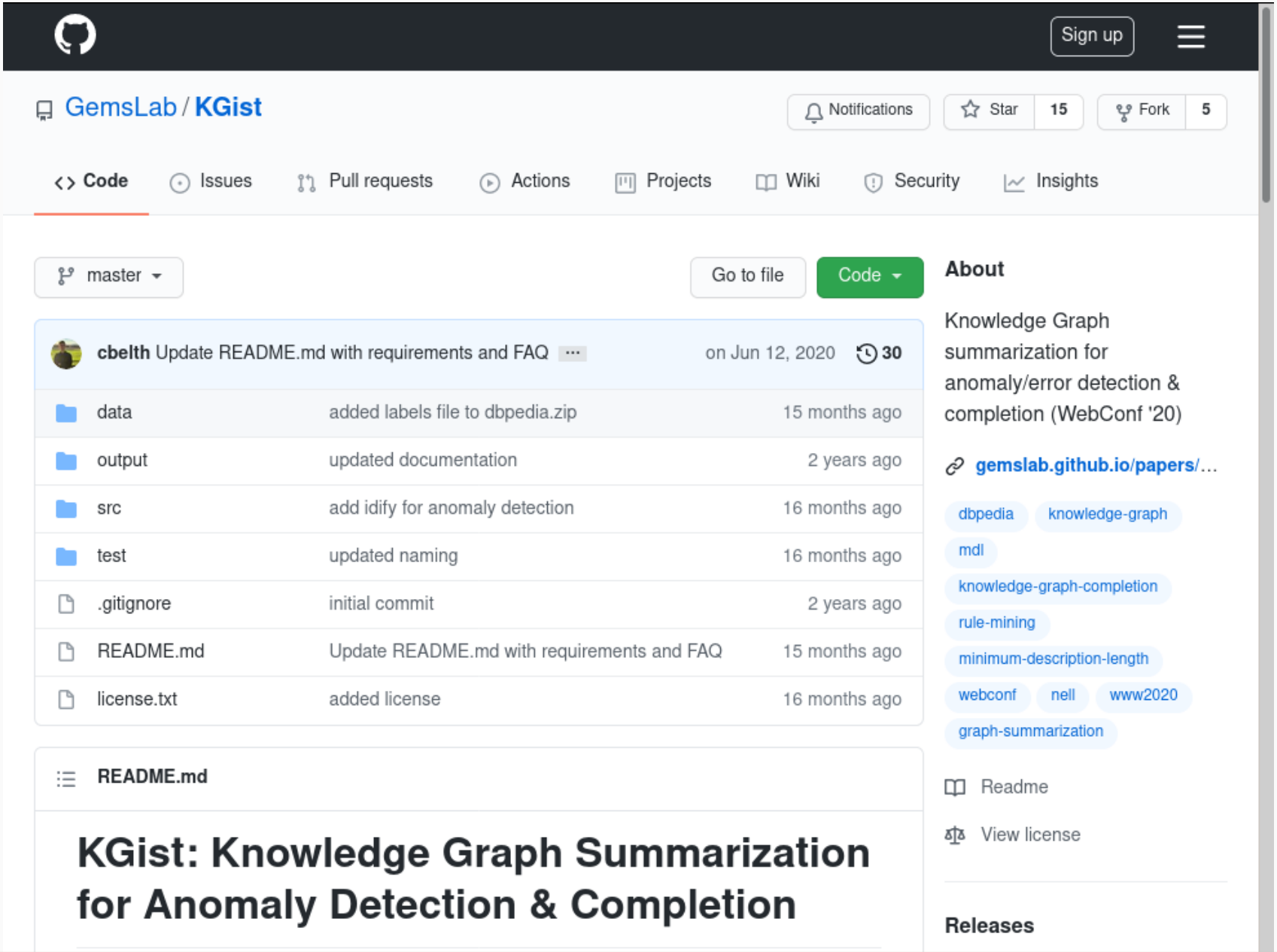
- access management
- file hosting
- wiki
- pull requests
- continuous integration
- ...

Around git

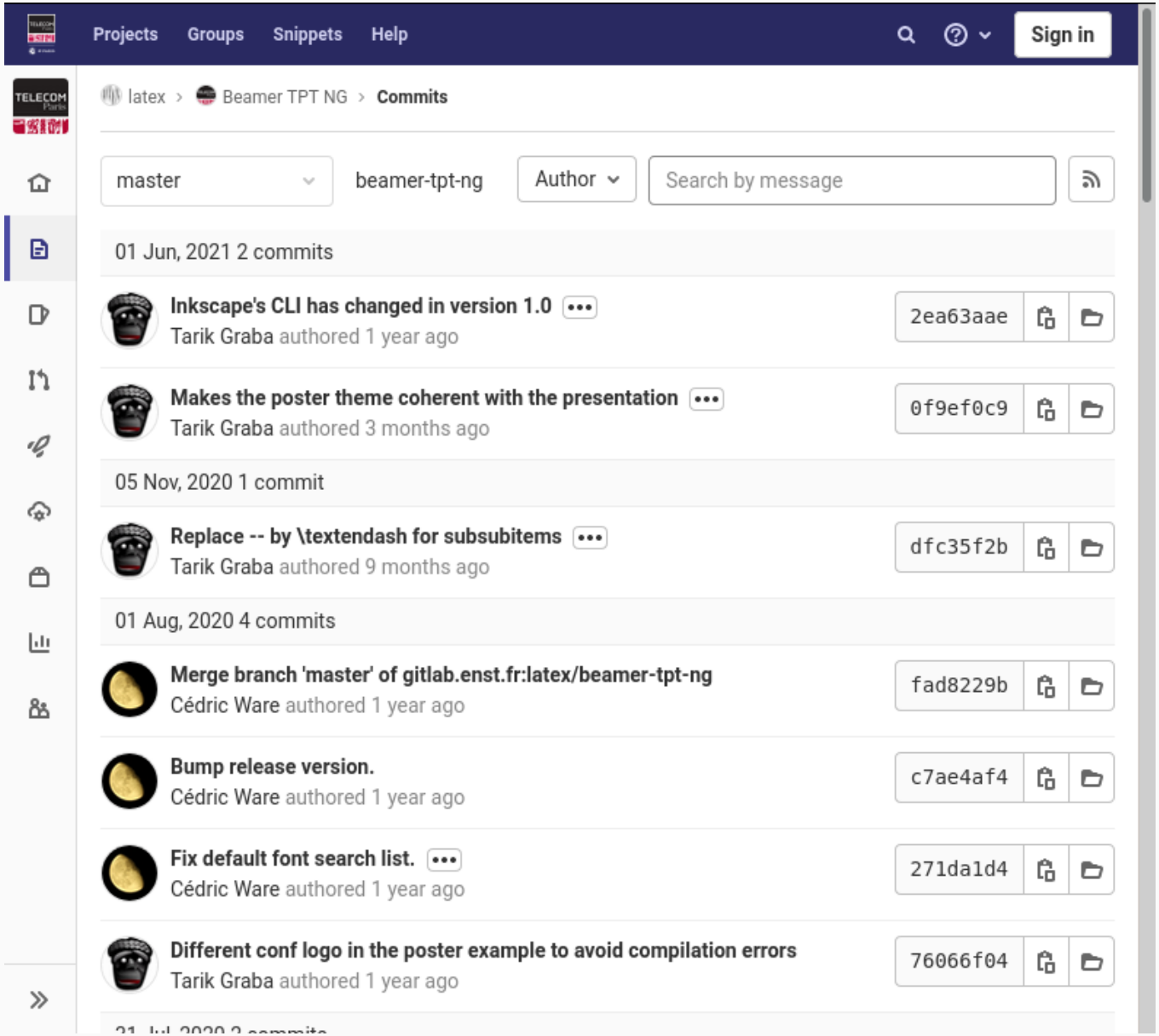
While Git works in a fully decentralized way but whole ecosystems have emerged around it with version control but also:

- access management
- file hosting
- wiki
- pull requests
- continuous integration
- ...

Github example



Gitlab example



Tools for computer scientists

SSH

SSH principles

Shell

Computers can be controlled with a *Graphical User Interface (GUI)* but historically they can also be controlled from the *Command line*.

Secure SHell (SSH)

Allow to get to the command line of another computers

SSH

SSH has many uses:

- Use computer from afar

  reserved for power users

- Run computation

  useful to run computation in e.g. lab rooms

- Move files

  useful to move files between the school system and your computer

- Protocol of connection (git)

  GIT often relies on ssh to "push" your changes

- Tunnel internet (much like a VPN)

  Access pay-walled articles from home

Good practices for programming

Good practices for programming

Writing code

IDE

There exists programming environments allowing you to:

- Have syntax coloring
- Automatic indenting or completion
- Shortcut for compilation/run and navigating through the errors
- ...

Don't write code on basic notepads!

Good practices for programming

Virtualenv

Package python

Advanced classes will often require you to install python packages. To avoid "polluting" your computer you can install them in virtualenv.

```
#creating the virtual env
virtualenv nameOfTheVirtualenv



# using the virtualenv
source nameOfTheVirtualenv/bin/activate #activation
which pip #check which pip we are using
pip install package name # install package in the venv
python # run python in the venv
jupyter-notebook # run jupyter equipped with the venv
```

Good practices for programming

Notebooks

Jupyter notebooks

jupyter Untitled (unsaved changes) Logout

File Edit View Insert Cell Kernel Widgets Help Not Trusted Python 3

Run Code

In [44]:
```
def make_numerical(df, cols):
    new_df = df.copy()
    for col in cols:
        new_df[col] = new_df[col].astype('category')
        new_df[col] = new_df[col].cat.codes
    return new_df
```

In [45]:
```
raw_penguins=pd.read_csv('penguins.csv', sep=',',header='infer')
```

In [46]:
```
penguins = raw_penguins. \
           dropna()
```

In [61]:
```
num_data = make_numerical(penguins,["island","species","sex"])
num_data
```

Out[61]:

	species	island	bill_length_mm	bill_depth_mm	flipper_length_mm	body_mass_g	sex	yea
0	0	2	39.1	18.7	181.0	3750.0	1	200
1	0	2	39.5	17.4	186.0	3800.0	0	200
2	0	2	40.3	18.0	195.0	3250.0	0	200
4	0	2	36.7	19.3	193.0	3450.0	0	200
5	0	2	39.3	20.6	190.0	3650.0	1	200
...	...	...	...	...	...	...	...	.
339	1	1	55.8	19.8	207.0	4000.0	1	200
340	1	1	43.5	18.1	202.0	3400.0	0	200

Jupyter notebooks

jupyter Untitled (unsaved changes) Logout

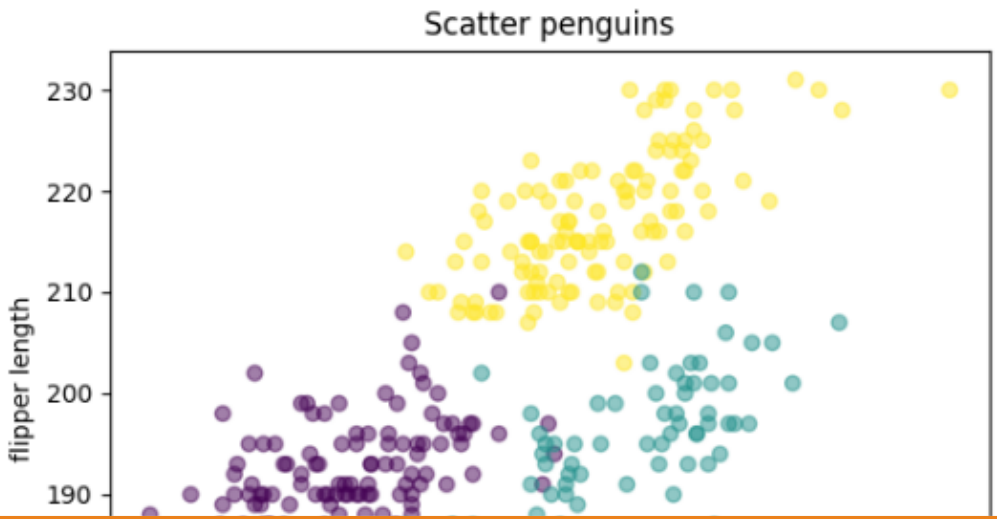
File Edit View Insert Cell Kernel Widgets Help Not Trusted Python 3

Run Code

Plotting bill length in function of flipper length (and show species)

In [62]:
```
plt.figure(get_plot())
plt.scatter(num_data['bill_length_mm'], num_data['flipper_length_mm'],
plt.title('Scatter penguins')
plt.xlabel('bill length')
plt.ylabel('flipper length')
plt.show()
```

```
<IPython.core.display.Javascript object>
```



Jupyter notebooks

Notebooks are great for:

- Test quickly code
- Manipulate data
- Visualize data
- Present your work

Overall it is great for "write-only" code.

Notebooks have HUGE limitations:

- You need to know the order in which the cells were launched
- Some piece of code might be deleted
- Hard to test
- Hard to version

Installing Jupyter notebooks

Notebooks are packaged in python:

```
#if using venv:
virtualenv ~/jupyterEnv
source ~/jupyterEnv/bin/activate


# installing them
pip install jupyter
jupyter-notebook
```

Good practices for programming

Writing code

Why write readable code ?

Generally, code is written once and read many times:

- by other people
  - peer-review,
  - homework correction,
  - inclusion in other people code
- also by yourself from the future
  - debugging,
  - restarting a paused code session,
  - copying code from the past,
  - etc.

Even when writing "write-only" code, it is a good exercise to train to write readable code.

How write readable code ?

The main tools to write readable code are:

- Use standardized code formatting
- Decompose your code into functions / modules
- Naming correctly variables and functions
- Comments

Code formatting

In general, be consistent on:

- indentation
- white spaces / tabulations
- capitalization
- style of functions calls
- style of comments

There are established conventions on code formatting

- see e.g. `https://pep8.org/` for python
- also there are tools to auto-format

Decomposing code

Writing code is all about taking a complex problem and dividing it recursively into subproblems.

Consequences:

- Don't reinvent the wheel each time
- Decompose your big projects into smaller projects
- Decompose your code into files
- Decompose your files into functions and classes
- Only a handful of variables should be global
- Files and functions should remain "small"

However, don't overdo this...

Naming functions / variables

Functions should explain what they do from a high level perspective:

- ☒ doStuff

Naming functions / variables

Functions should explain what they do from a high level perspective:

- ☒ doStuff
- ☑ ComputeVAT

Naming functions / variables

Functions should explain what they do from a high level perspective:

- ☒ doStuff
- ☑ ComputeVAT
- ☒ SumMultiplyCategoryToPrices

Naming functions / variables

Functions should explain what they do from a high level perspective:

- ☒ doStuff
- ☑ ComputeVAT
- ☒ SumMultiplyCategoryToPrices
- ☑ cleanAbort
- ☑ loadDataset
- ☒ readFile

Variables should explain what they store:

- ☒ ff

Naming functions / variables

Functions should explain what they do from a high level perspective:

- ☒ doStuff
- ☑ ComputeVAT
- ☒ SumMultiplyCategoryToPrices
- ☑ cleanAbort
- ☑ loadDataset
- ☒ readFile

Variables should explain what they store:

- ☒ ff
- ☑ nbFalsePositives

Naming functions / variables

Functions should explain what they do from a high level perspective:

- ☒ doStuff
- ☑ ComputeVAT
- ☒ SumMultiplyCategoryToPrices
- ☑ cleanAbort
- ☑ loadDataset
- ☒ readFile

Variables should explain what they store:

- ☒ ff
- ☑ nbFalsePositives
- ☒ numberOfWrongItems, GoodItemsCount (choose one)

Naming functions / variables

Functions should explain what they do from a high level perspective:

- ☒ doStuff
- ☑ ComputeVAT
- ☒ SumMultiplyCategoryToPrices
- ☑ cleanAbort
- ☑ loadDataset
- ☒ readFile

Variables should explain what they store:

- ☒ ff
- ☑ nbFalsePositives
- ☒ numberOfWrongItems, GoodItemsCount (choose one)

Comments

A good comment is one that:

- provides some documentation on the code
- explain the structure of the code
- provides insights on some part of the code
- explain a piece of code from a high-level perspective
- Does not repeat the code

Comments

A good comment is one that:

- provides some documentation on the code

  This function expects two integers x and y and returns $x \times y$

- explain the structure of the code

  Here we load the data, here we train the model, etc.

- provides insights on some part of the code

  x+=1 # This accounts for the top square

- explain a piece of code from a high-level perspective

  Here we try to pre-compute the distance of all points to A

- Does not repeat the code

  computeShortestPath() # will compute shortest path

Is it useful for masters?

You will mainly have write-once read-once code this year but you should start using good practices because writing does not take more time and you will need it someday!

Good practices for programming

Testing your code

Tests

Unit tests

Unit tests tries some very specific piece of code to make sure it works ok.

e.g. test each function or each member of each class

Functional tests

Runs the whole program and make sure the output is what was expected.

easy to set up for programs without interfaces

Regression tests

Each time a bug is discovered, we introduce tests to make sure this bug is covered.

Example of unittest package in Python

See https://docs.python.org/3/library/unittest.html

```python
import unittest

class TestStringMethods(unittest.TestCase):

    def test_upper(self):
        self.assertEqual('foo'.upper(), 'FOO')

    def test_isupper(self):
        self.assertTrue('FOO'.isupper())
        self.assertFalse('Foo'.isupper())

    def test_split(self):
        s = 'hello world'
        self.assertEqual(s.split(), ['hello', 'world'])
        # check that s.split fails when the separator is not a string
        with self.assertRaises(TypeError):
            s.split(2)

if __name__ == '__main__':
    unittest.main()
```

Testing in practice

Not using framework to test your code is unthinkable for professionals however for a short piece of code for a class you can test manually:

- check that the output has the correct format
- check on small examples it makes sense
- test the output is reasonable
- if possible run manually a few functional tests before submitting your work

Exercises!

Exercises

- Clone the following repository and create a short presentation: `https://gitlab.telecom-paris.fr/latex/beamer-tpt-ng`
- Read the pep8 (`https://pep8.org/`) and read and do some of the exercises here `http://web.mit.edu/6.005/www/fa15/classes/04-code-review/`
- Create a repository on Télécom's gitlab (use the sign-in with Shibboleth and use your Télécom's account
- Install a password manager (e.g. KeepassX)
- Start doing backups
- Make sure your disk is encrypted (if not consider doing it someday)
- Write yourself a résumé using LaTeX (see `https://www.overleaf.com/latex/templates/tagged/cv` for templates