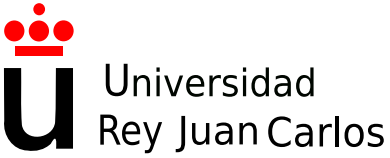
How to Do Your Own Analysis of the Kernel Development

Jesus M. Gonzalez-Barahona

@jgbarah
Bitergia
GSyC/LibreSoft, Universidad Rey Juan Carlos

Edinburgh, UK, October 23rd 2013
(draft, work in progress)



Bitergia: analytics for your peace of mind

Started operations in July 2012
Builds on the experience of LibreSoft R&D group
Offering professional products and services
Focused on:

- Metrics dashboards about software development (including community metrics)
- Specific studies and reports (based on metrics and facts collection)

http://bitergia.com
http://blog.bitergia.com

Free software is (in many cases) special

Source code available

Open development model (usually)

- Many details about the internals of the development process
- Intense use of tools for coordination
- Lots of information is tracked, and available

Developers & users communities are important

- sustainability
- pooling of resources
- innovation

Measuring, measuring, measuring

Information about code, community, development can be retrieved, organized, analyzed

Metrics-tracked processes

Quantitative, objective data: facts, not opinions
Specific questions can be answered
Several areas of interest:

- Developers:
  Understanding, improving development processes
  Early detection of potential problems, bad smells
- Community:
  General activity, contributions
  Long-term sustainability, evolution, reaction to issues

But data has to be extracted, mined...

Data lives in repositories not always designed to release all their data easily:

tools are needed to retrieve and extract it

Data includes many complexities and details

tools are needed to filter, organize it

But data has to be analyzed, visualized...

Casual observation is not enough, analysis is needed:

tools are needed
for statistical and other kinds of analysis

Analysis is not enough, visualization may help:

tools are needed for interactive visualization

The MetricsGrimoire approach

Set of tools specialized in retrieving information from different kinds of repositories. Among them:

- CVSAnalY: source code management (CVS, Subversion, git, etc.)
- Bicho: issue tracking systems (Bugzilla, Jira, SourceForge, Allura, Launchpad, etc.)
- MLStats: mailing lists (mbox files, Mailman archives, etc.)

Store all the information in SQL databases with similar structure

`http://metricsgrimoire.github.io`

The vizGrimoire approach

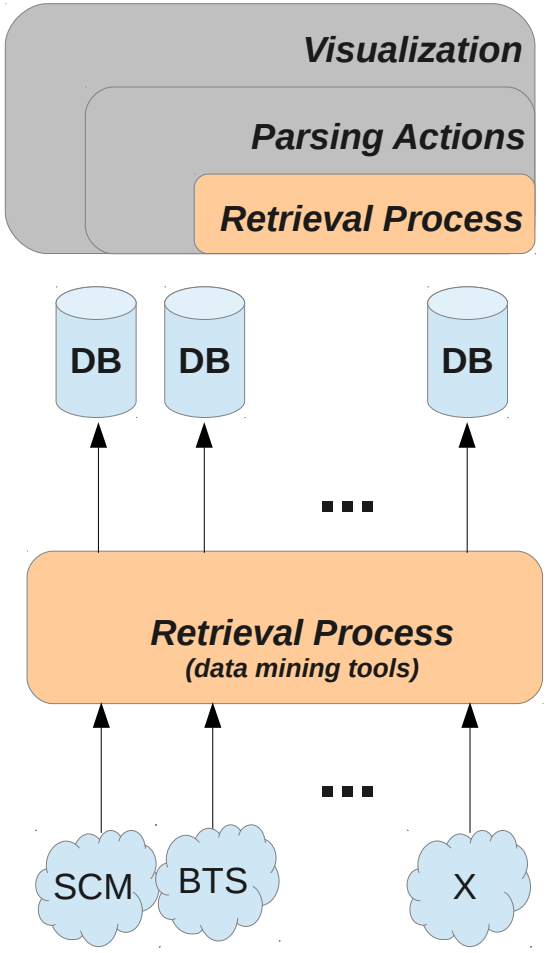
Set of tools for analyzing and visualizing data produced by MetricsGrimoire:

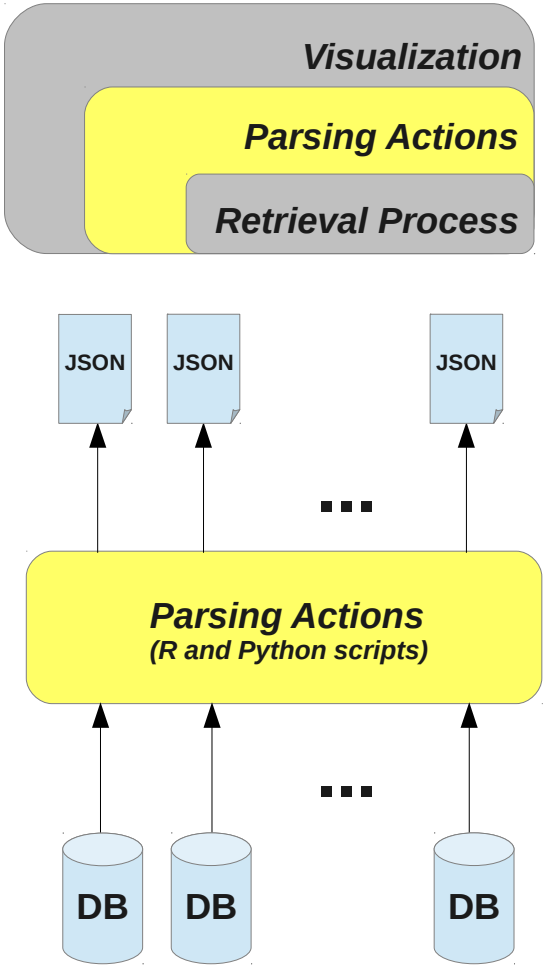
- vizGrimoireR: R package for analysis and producing JSON files
- vizGrimoireJS: JavaScript library for visualizing JSON files
- Several dashboards: based on vizGrimoireJS

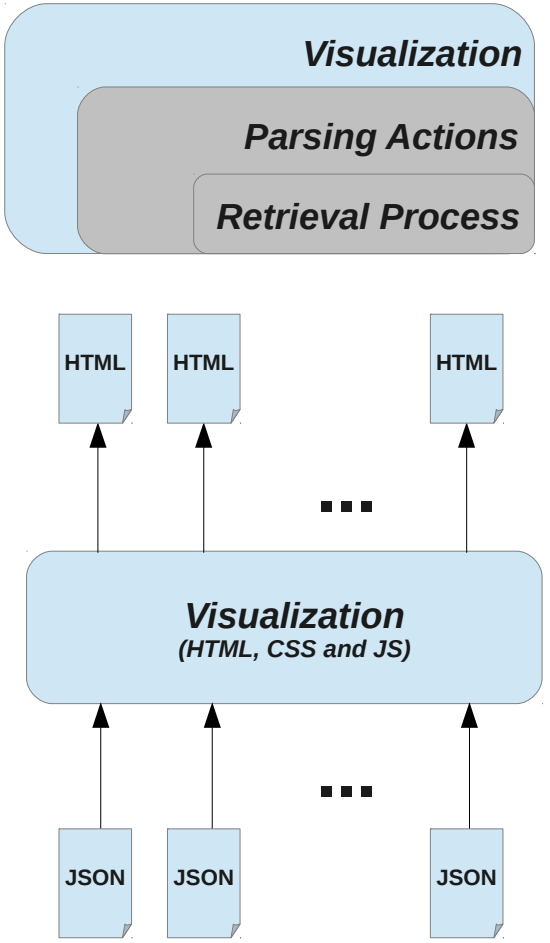
From SQL to JSON to visualization

`http://vizgrimoire.github.io`

The *Grimoire combination







MetricsGrimoire: CVSAnalY

- Browses an SCM repository producing a database with:
  - All metainformation (commit records, etc.)
  - Metrics for each release of each file
- Also produces some tables suitable for specific analysis
- Multiple SCMs: CVS, svn, git (Bazaar, Mercurial through git)
- Whole history in the database, it's possible to rebuild the files tree for any revision
- Tags and branches support
- Option to save the log to a file while parsing
- Extensions system, incremental capabilities
- Multiple database system support (MySQL and SQLite)

MetricsGrimoire: CVSAnalY extensions

- Extension: a "plugin" for CVSAnalY
- Add information to the database, based in the information in the database and maybe the repository
- Usually: new tables for specific studies
- Simple example: commits per month per commiter
- Extensions add one or more tables to the database but they never modify the existing ones

MetricsGrimoire: CVSAnalY extensions

Some examples:

- FileTypes: adds a table containing information about the type of every file in the database (code, documentation, i18n, etc.)
- Metrics: analyzes every revision of every file calculating metrics like sloc and complexity metrics (mccabe, halstead). It currently supports metrics for C/C++, Python, Java and ADA.
- CommitsLOC: adds a new table with information about the total lines added/removed for every commit

MetricsGrimoire: Bicho

Parsing issue tracking systems
Results stored in a MySQL database
Information about each issue (ticket), and its modifications
Currently it supports:

- SourceForge (HTML parsing)
- Bugzilla: GNOME, KDE, others
- Jira, Allura, Launchpad, GitHub, RedMine (API)

Incremental
Supports Gerrit as well (code review)

MetricsGrimoire: MailingListStats

Parses mbox information (RFC 822)
Deals with Mailman archives
Stores results (headers, body) in a MySQL database:

- Sender, CCs, etc.
- Time / Date
- Subject
- ...

Incremental
Can store multiple projects in a single database

vizGrimoireR & vizGrimoireJS

Once information is retrieved, and is in a format suitable for querying:

- it can be queried directly in the database
- it can be analyzed from R
- it can be filtered, manually inspected, improved
- it can be combined, cross-analyzed
- it can be visualized

Milking the databases, producing dashboards

Now...

...howto for the Linux kernel

Data sources used

Git snapshot with changes since 1992,
by Yoann Padioleau

- reliable about individual changes since 2002)
- updated (git pull) up to September 2013

http://archive.org/details/git-history-of-linux
git://git.kernel.org/pub/scm/linux/kernel/git/torvalds/linux.git

Gmane linux-kernel mailing list

http://dir.gmane.org/gmane.linux.kernel

Running the tools: getting MetricsGrimoire

Get source code from GitHub repositories and follow instructions

...or...

run metricsgrimoire-setup.py

https://github.com/VizGrimoire/VizGrimoireR/blob/master/misc/metricsgrimoire-setup.py

Running the tools: CVSAnalY (git)

- Get the git repo with historic information
- Run CVSAnalY to populate cvsanalydb database

```
wget http://archive.org/download/
  git-history-of-linux/full-history-linux.git.tar
tar xvf full-history-linux.git.tar
cd full-history-linux
git pull
[Create databases cvsanalydb]
cvsanaly2 -u user -p XXX -d cvsanalydb \
  --extensions=FileTypes,CommitsLOC .
[From vizGrimoireR/misc, >5,000 dup ids found]
unifypeople.py -u user -p XXX -d cvsanalydb
```

Running the tools: MLStats (mailing list)

```
[From vizGrimoireR/examples/linux]
get-gmane-archive.sh
mlstats --db-user=user --db-password=XXX \\
  --db-admin-user=adminuser --db-admin-password=XXX \\
  --db-name=mlstatsdb \\
  mail_dir
```

Running the tools: Producing a basic dashboard

Installation of vizGrimoireR as package for R

https://github.com/VizGrimoire/VizGrimoireR/wiki

Produce all the files for the dashboard

```
[ in vizGrimoireR directory, run R scripts to produce JSON ]
vizGrimoireJS/run_scripts-linuxkernel.sh
[ get HTML (JavaScript, HTML, CSS) files for dashboard ]
git clone git@github.com:VizGrimoire/VizGrimoireJS.git \
  linux-dashboard
cd linux-dashboard
git checkout linux
[ copy JSON files produced above ]
cp ../json/* ./data/json
```

Export the directory via HTTP

Access it your favorite web browser

Basic dashboard

Linux Kernel () Summary Source Code Mailing Bitergia

464652 commits 10914 developers 1560736 mail messages

Code Developers

10914

Core	Regular	Occasional
690	1804	8420

464652 Commits

-43 (41%) 7 Days Change | -2419 (78%) 30 Days Change | -962 (1%) 365 Days Change

10914 Authors

-15 (31%) 7 Days Change | -393 (59%) 30 Days Change | 43 (1%) 365 Days Change

Discussion Participants

28602

Thread Initiators	First Repliers
24431	16208

1560736 Sent

-1248 (25%) 7 Days Change | 2266 (14%) 30 Days Change | 29420 (17%) 365 Days Change

28602 Senders

-41 (7%) 7 Days Change | 81 (8%) 30 Days Change | -152 (3%) 365 Days Change

Project name: Linux Kernel

Time period for the different data sources:
- git: 1999-01-01 to 2013-10-20
- Mailing lists: 1999-04-09 to 2013-10-21

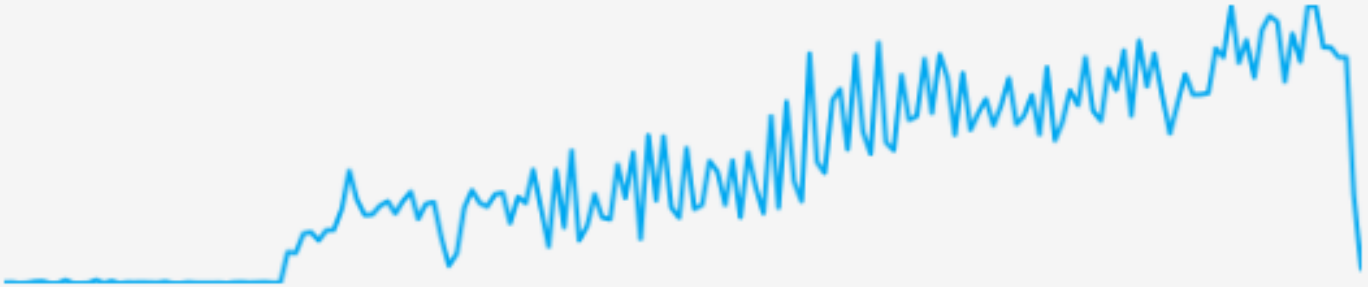
MySQL database dumps with the complete retrieved datasets
JSON files serving the data shown in the plots

Limited functionality, work in progress

http://bitergia.com/public/previews/2013_10_linux/browser

Basic dashboard (git)

Linux Kernel () Summary Source Code Mailing Bitergia



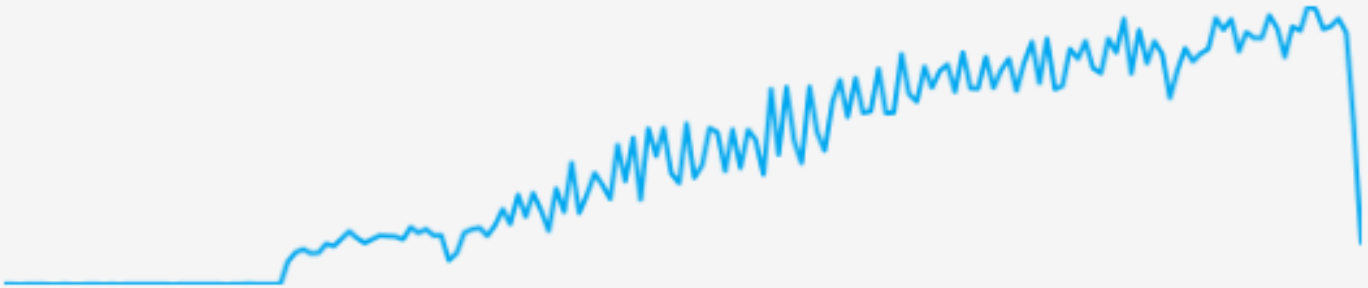
464652
Commits

-43 (41%) 7 Days Change
-2419 (78%) 30 Days Change
-962 (1%) 365 Days Change

42.5739 avg. com
author

Top Authors

	commits
Linus Torvalds	19797
Len Brown	9430
David S. Miller	8150
Greg KH	5005
Mark Brown	4598
Russell King	4586
Takashi Iwai	4353
Ingo Molnar	4050
Steve Grubb	3189
Tejun Heo	3105

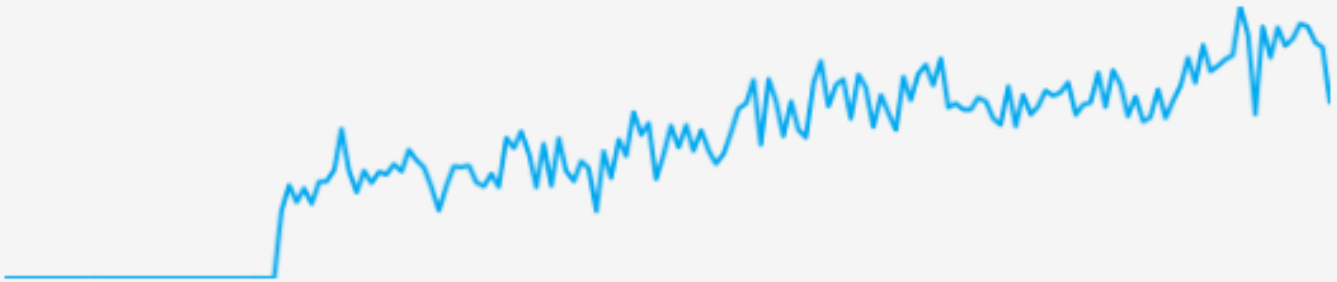


10914
Authors

-15 (31%) 7 Days Change
-393 (59%) 30 Days Change
43 (1%) 365 Days Change

Basic dashboard (mailing list)

Linux Kernel () Summary Source Code Mailing Bitergia



1560736
Sent

4871 (-25%)
7 Days Change

15496 (+14%)
30 Days Change

171831 (+17%)
365 Days Change

28602
Senders

295 (+118%)
7 Days Change

492 (+173%)
30 Days Change

2356 (+98%)
365 Days Change

A more complete dashboard: MediaWiki

Summary | Source Code | Issues | Mailing | Code Review | IRC | Top Contributors | Focused analysis

Reference Card

Project name: Wikimedia

Changes to source code: git
- Time period: 2003-04-14 2013-10-21
- Commits: 226857
- Committers: 612
- Authors: 683

Tickets bugzilla
- Time period: 2004-08-10 2013-10-21
- Issues: 41935
- Openers: 4936

Messages (mailman): Mailing lists
- Time period: 2002-02-08 2013-10-21
- Total messages: 156833
- Senders: 7440

MySQL database dumps with the complete retrieved datasets
JSON files serving the data shown in the plots

226857 Commits

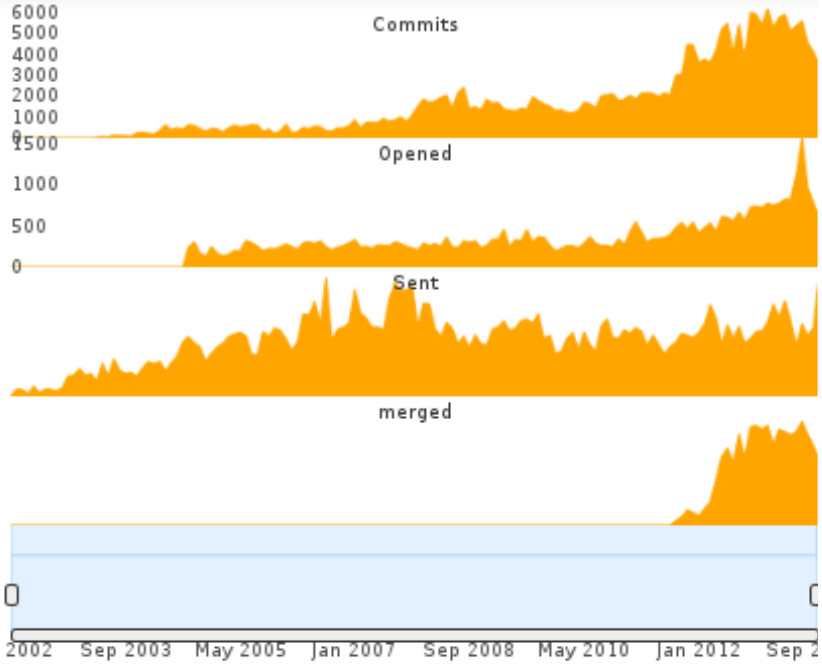
week:1104 (1%) *month*:4723 (8%) *year*:64878 (21%)

41935 Opened

week:173 (-6%) *month*:820 (-3%) *year*:10716 (65%)

156833 Sent

week:287 (-5%) *month*:2954 (153%) *year*:18205 (18%)



Top Authors

	commits
raymond	21920
Siebrand Mazeland	20914
L10n-bot	18940
Leslie Carr	8670
Brion Vibber	7803

Top Openers

	opened
sam	757
Brion Vibber	580
Liangent	563
Nikerabbit	541
bugzilla.wikimedia	513

Top Senders

	sent
Brion Vibber	8610
noreply	6777
Platonides	4665
xqt	2886
Simetrical+wikilist	2628

http://korma.wmflabs.org/

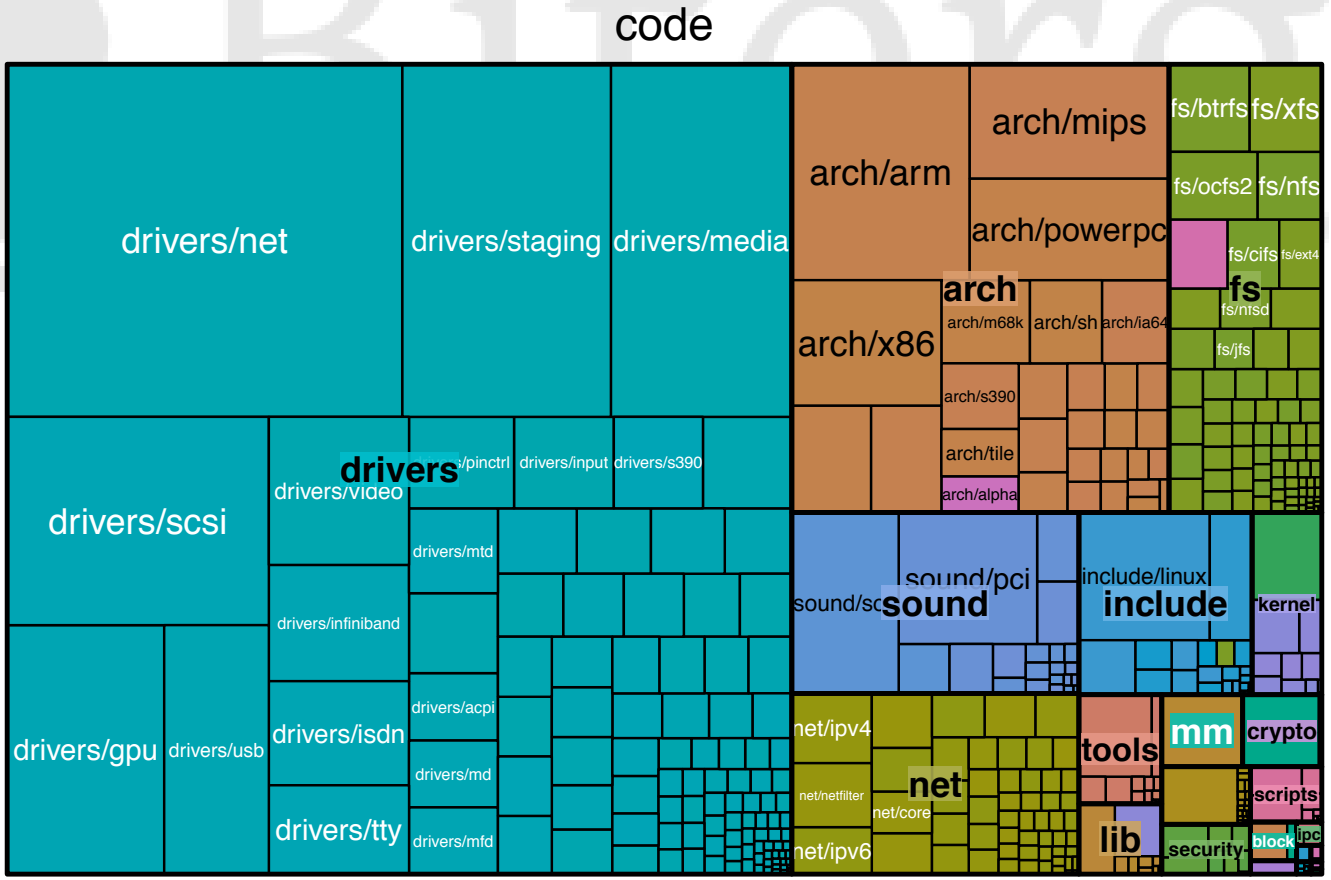
Case: Activity per directory in git repository

Three CVSAnalY tables involved:

- scmlog: metadata for each commit (author, date, ...)
- actions: metadata for each action on a file (file, commit, ...)
- file_links: metadata for each file (name, path, ...)

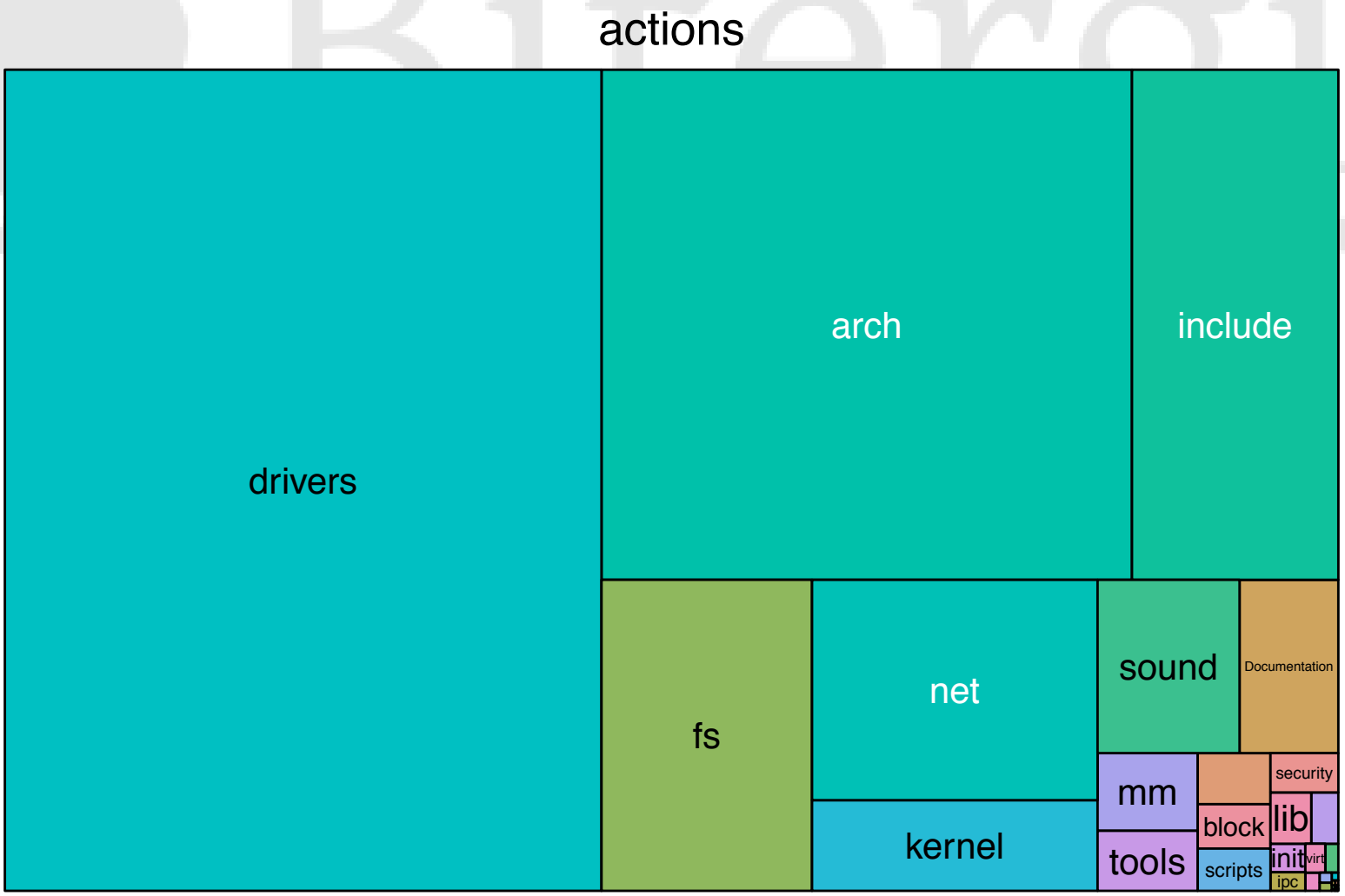
Group all actions involving files under a certain subdirectory,
calculate number of actions and (distinct) authors involved

Size (LOC) per subdirectory

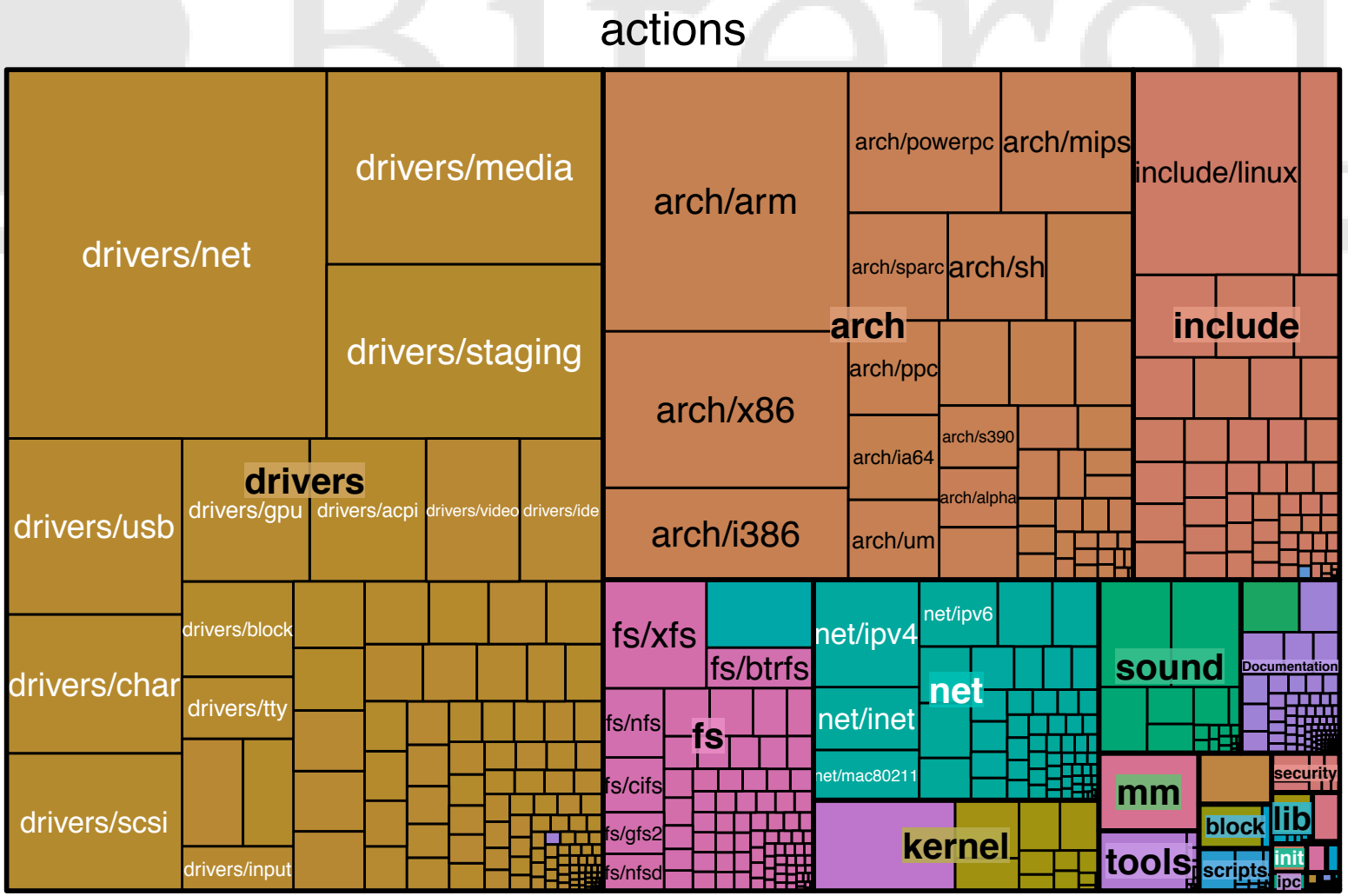


cloc –csv –skip-uniqueness –by-file

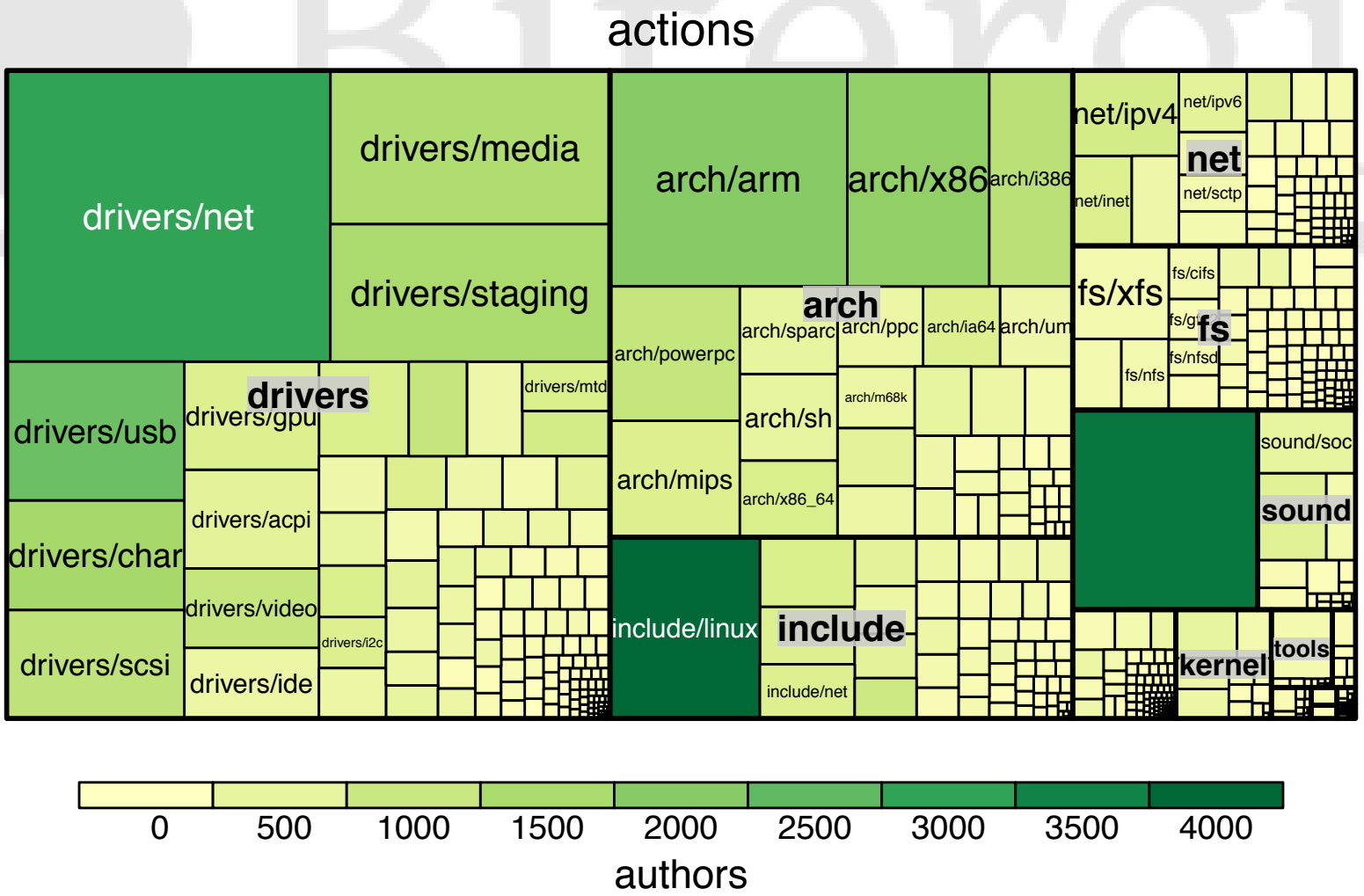
Changes per directory



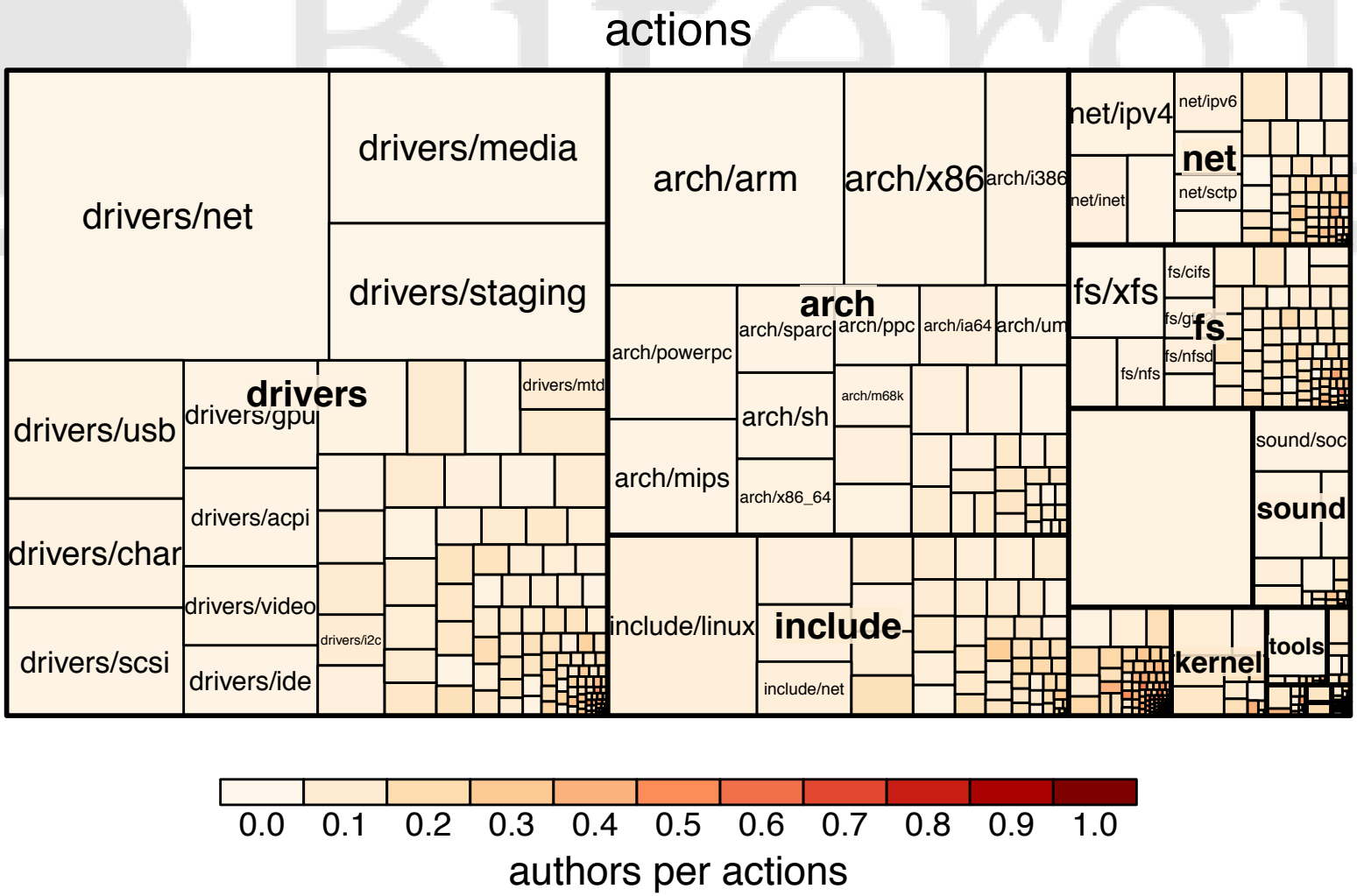
Changes per directory: more detail



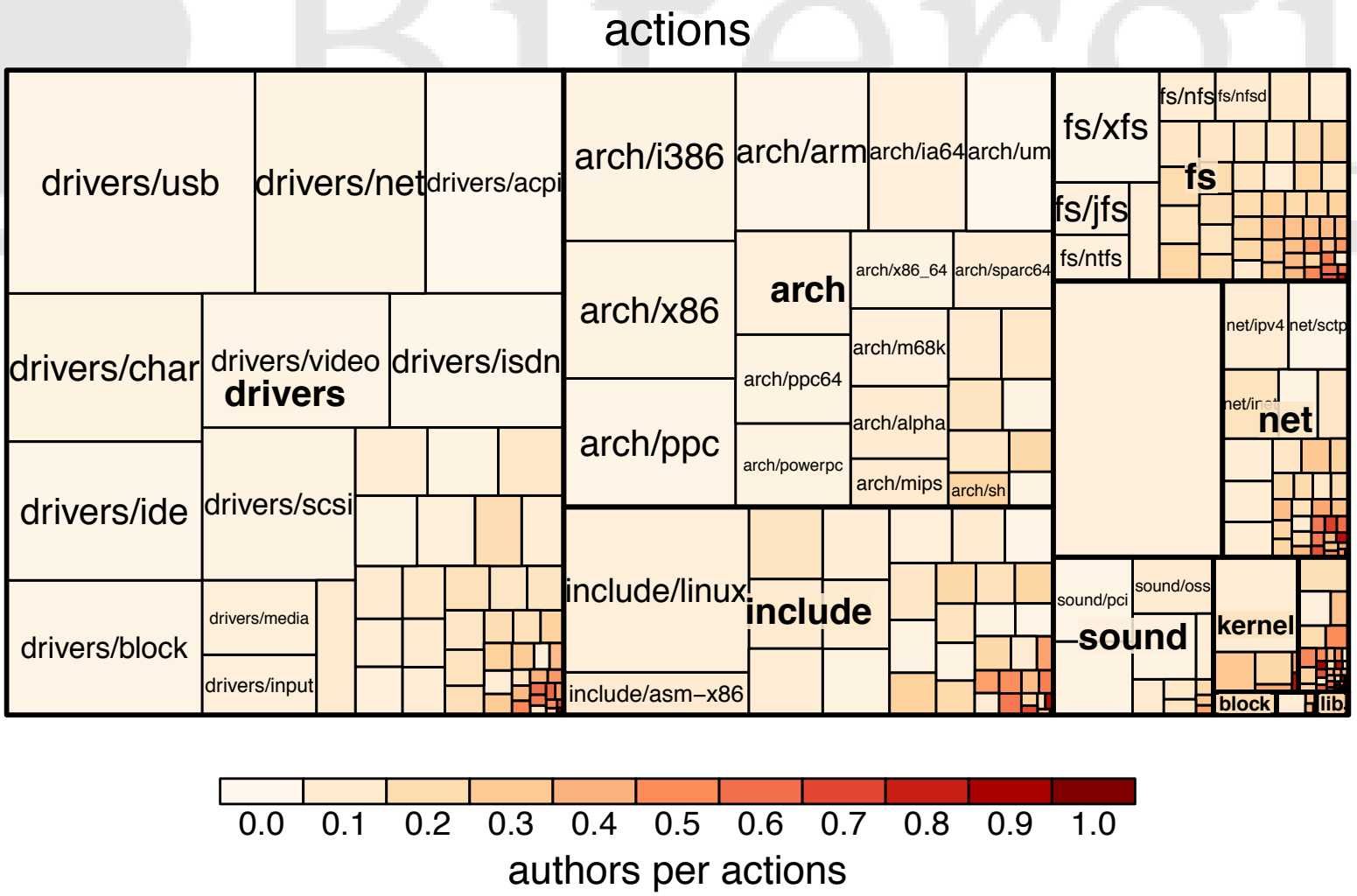
Changes per directory: color is number of authors



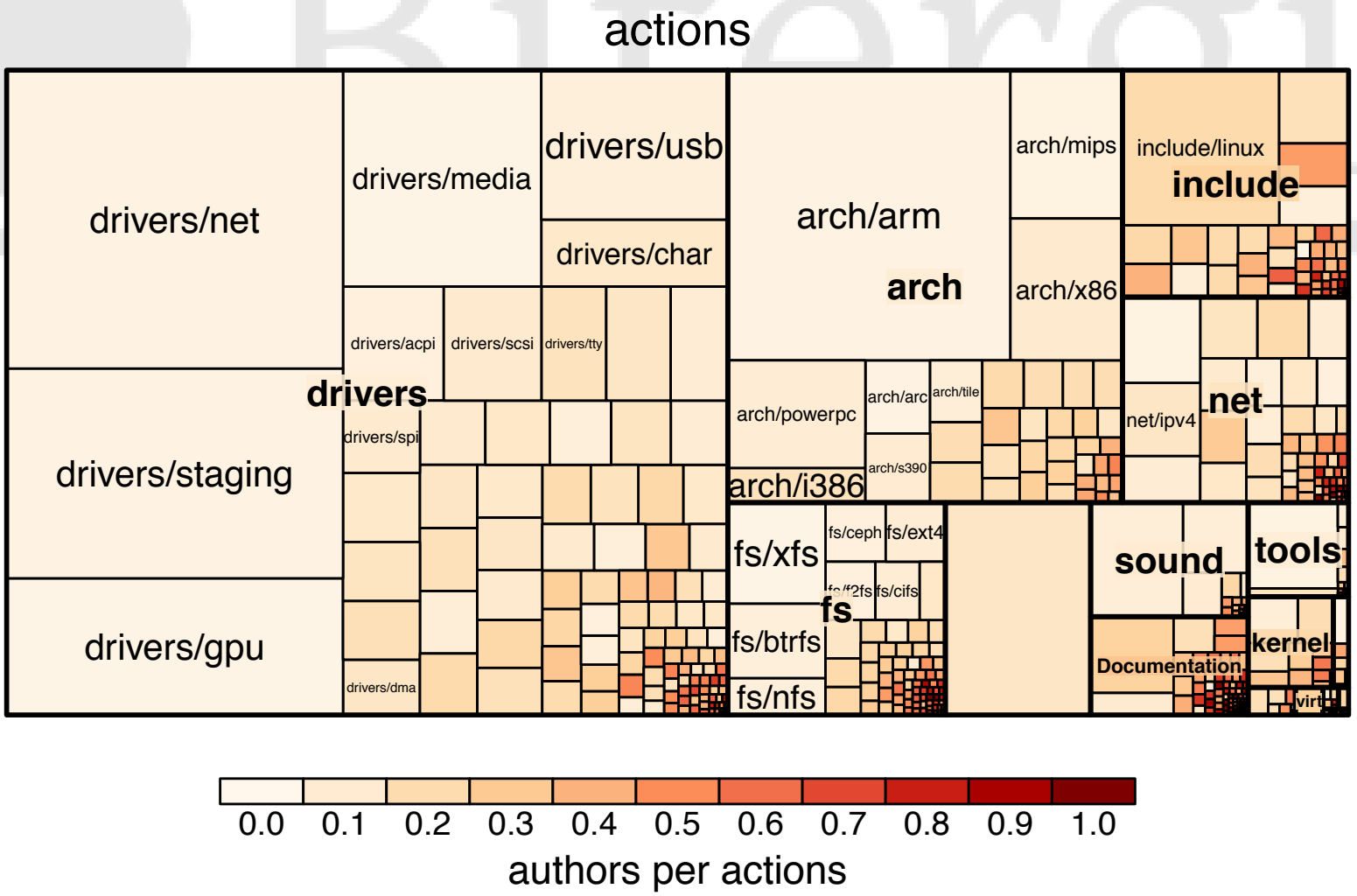
Changes per directory: color is density of authors



Changes per directory: 2002



Changes per directory: 2013



Case: experience of developers

Different ids for same developers have to be merged

- simple heuristics
- $> 5,000$ dup ids for about 10,000 devels

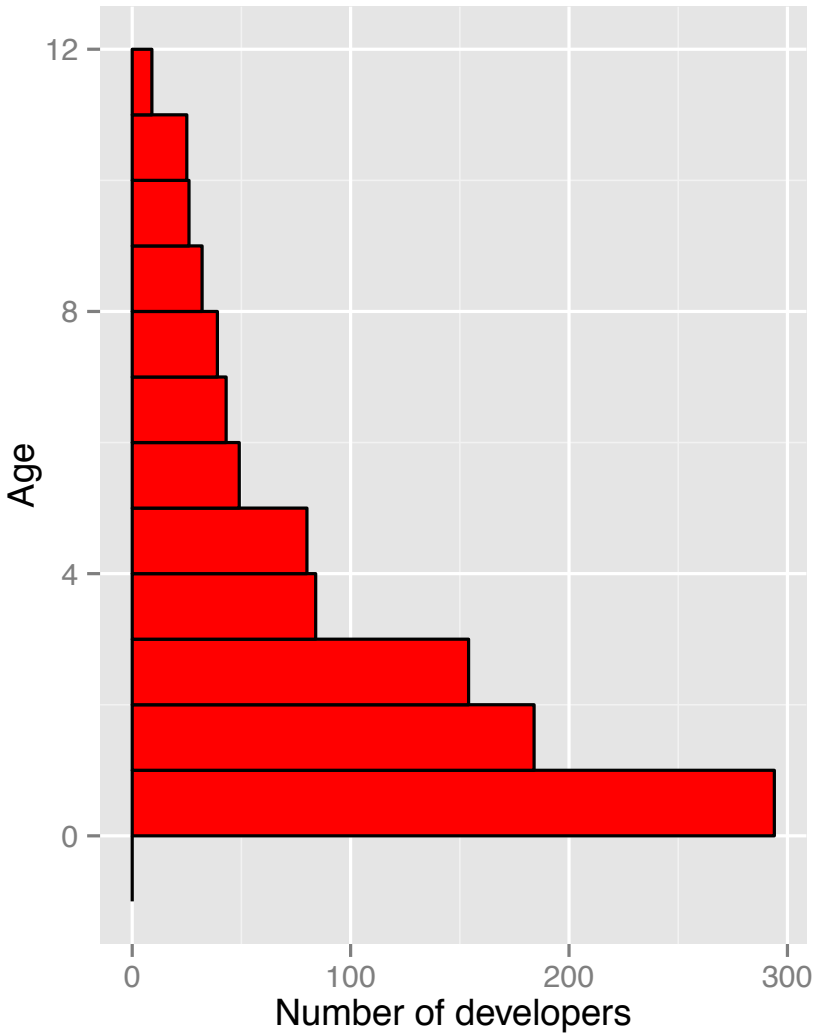
`https://github.com/VizGrimoire/VizGrimoireR/blob/master/misc/unifypeople.py`

Three CVSAnalY tables involved:

- scmlog: metadata for each commit (author, date, ...)
- people: metadata for ids corresponding to developers
- upeople: 'unified' ids for developers

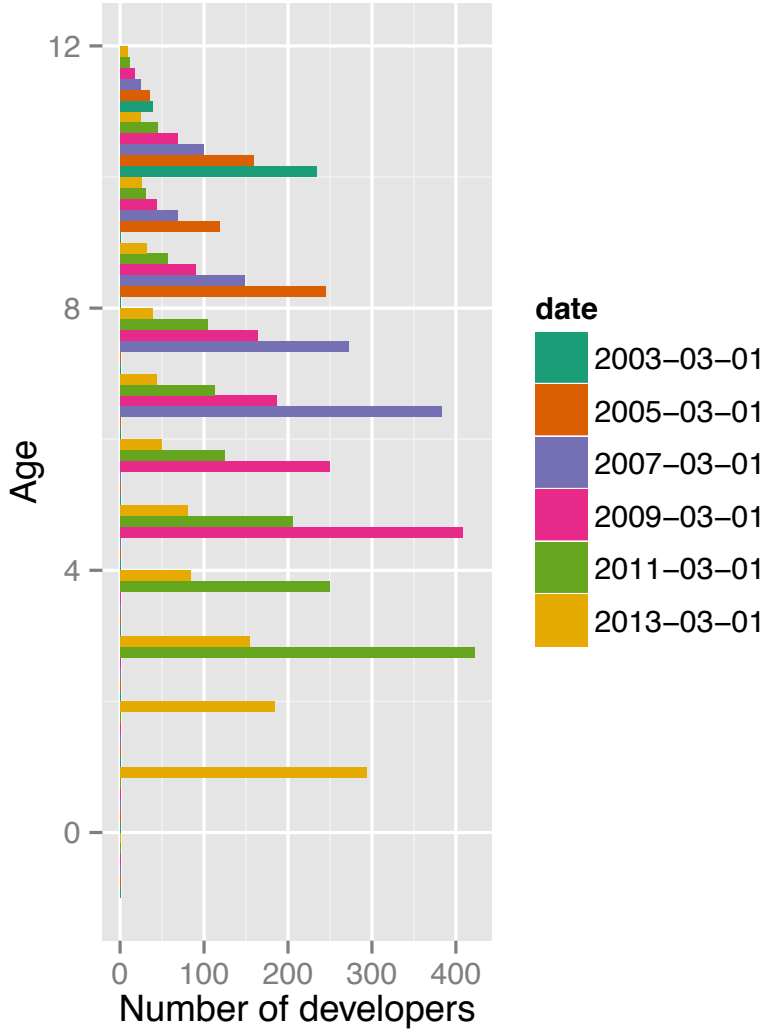
Calculate 'age in project' for active developers at a certain spot in time, grouping them by 'generations'

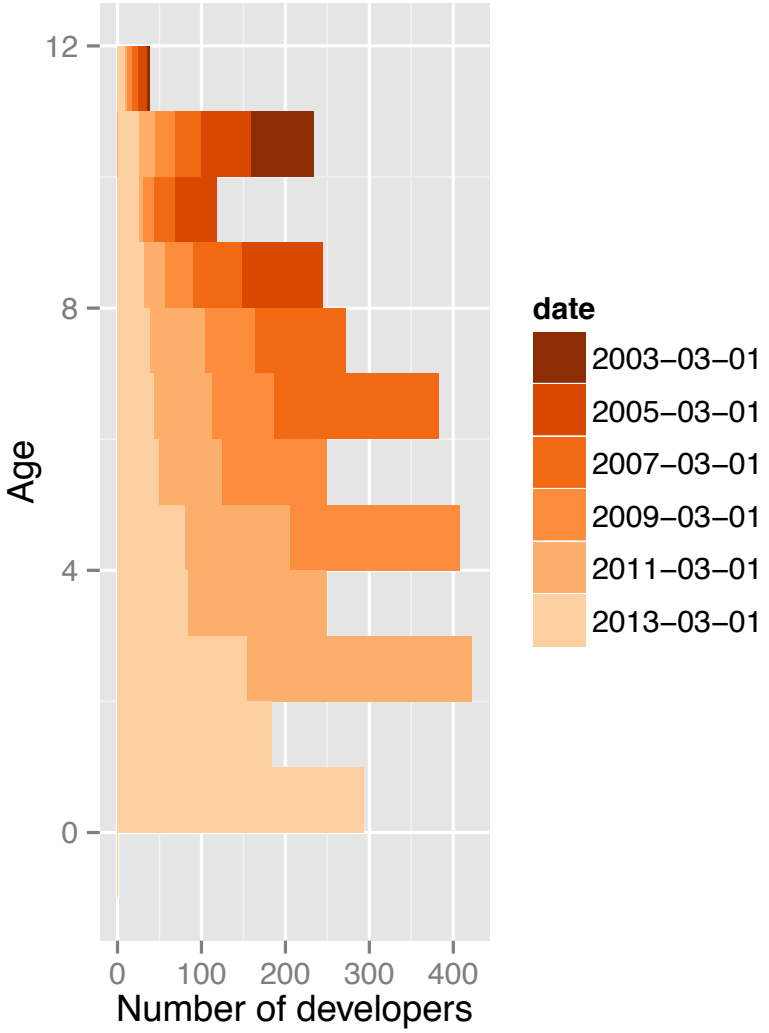
Experience: demography prydamid (git)



Generations (active authors of commits), March 1st, 2013

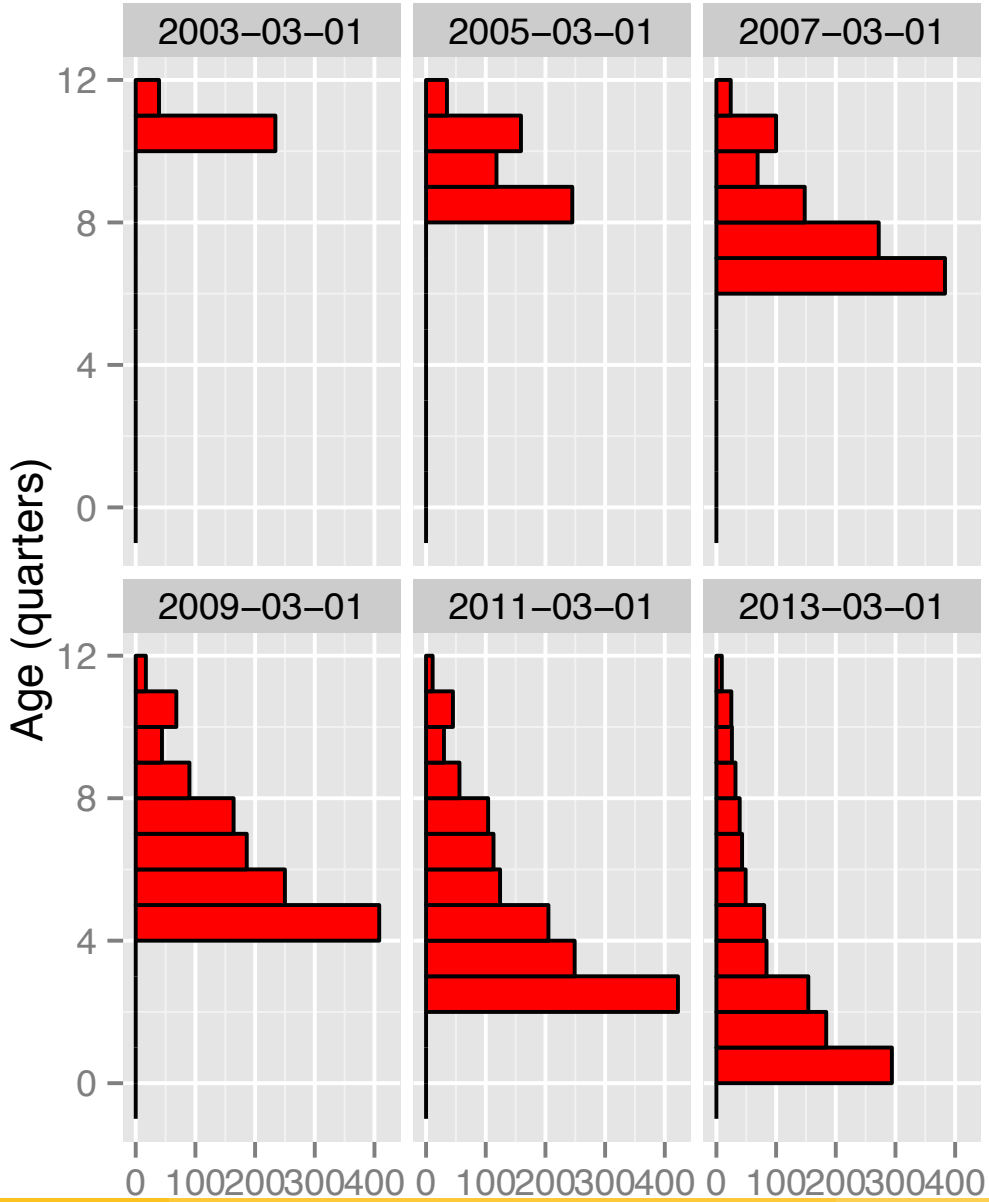
Experience: demography prydamid (git)



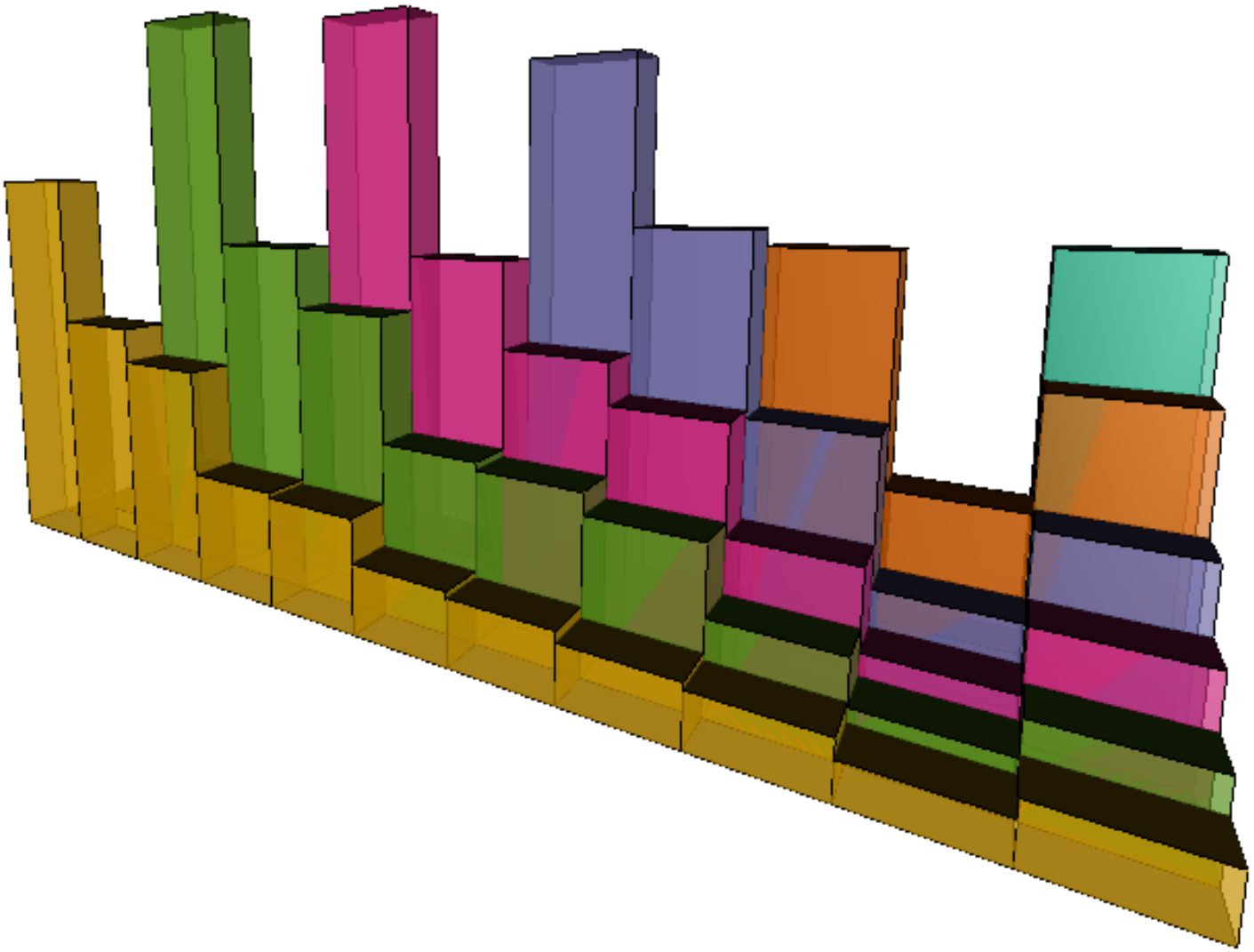


Comparison for pyramids every two years

Experience: demography prydamid (git)



Experience: demography prydamid (git)



3D version of pyramids every two years

Case: where do developers work?

Study on mailing list:

- assumption: time zones of mail tools are correct
- time zones correspond roughly to geographical areas

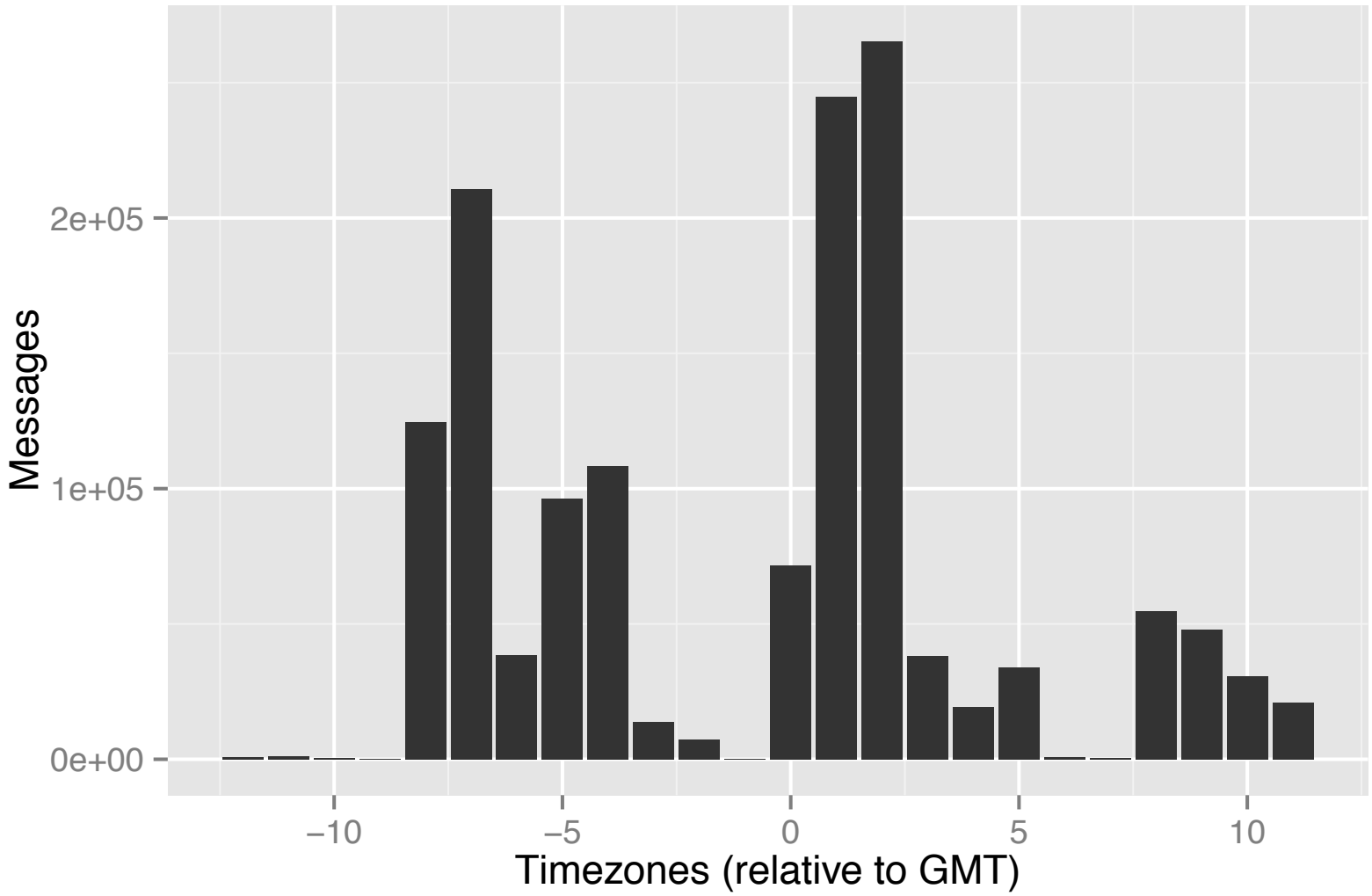
http://en.wikipedia.org/wiki/Time_zone#UTC_offsets_worldwide

- Americas: GMT-8 to GMT-2 (US/Canada: -8 to -4)
- Europe/Africa/Middle East: GMT to GMT+5
- East Asia/Australia: GMT+8 to GMT+11

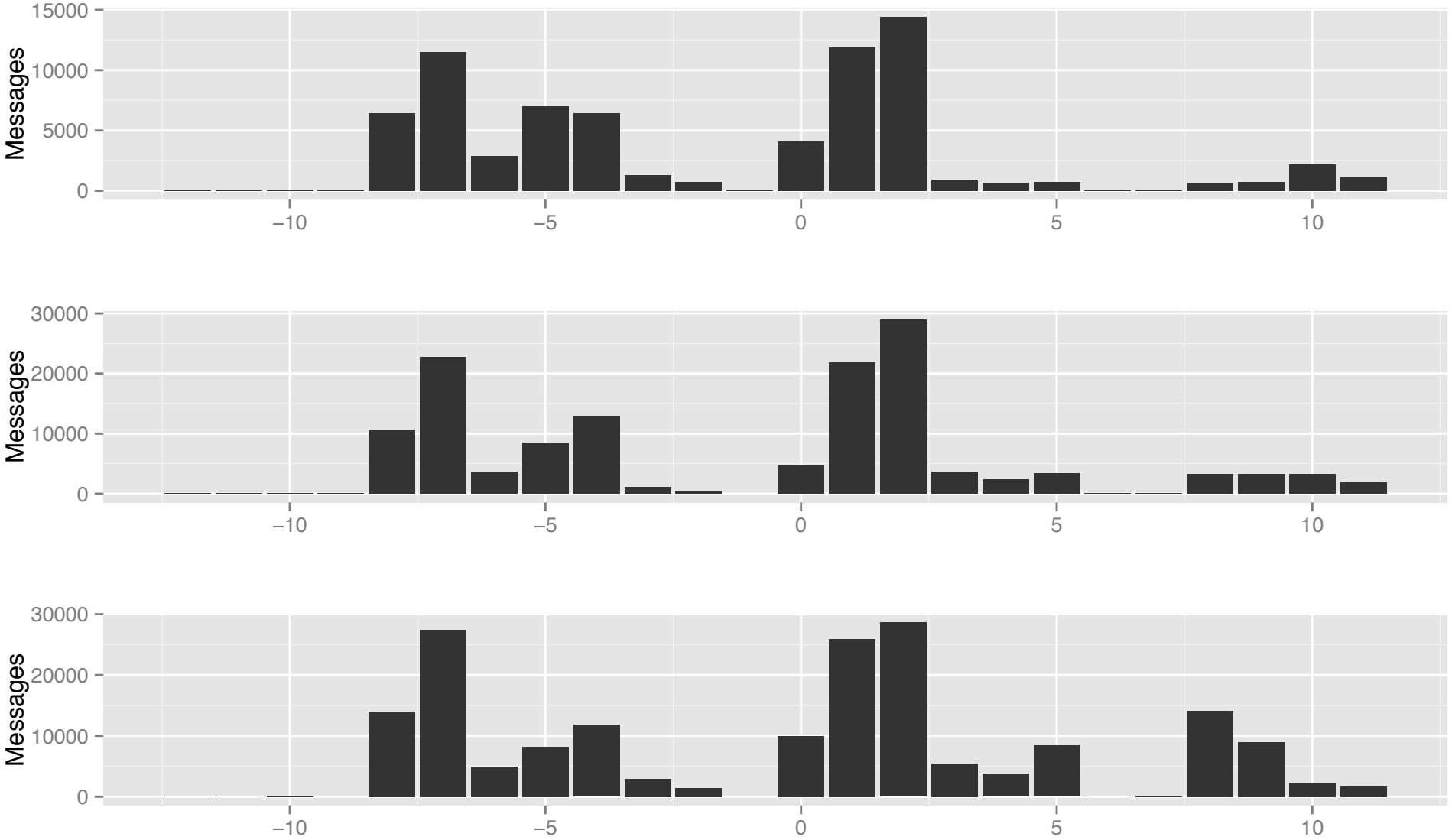
All the info in one MLStats table:

- messages: main headers of each message

Timezone origin of messages



Timezone origin of messages (2002, 2007, 2012)



Takeaways

Run MetricsGrimoire on the repositories,
get your own databases

Use our databases,
use vizGrimoire to run your own analysis,
produce your own visualizations

Use our analysis scripts,
modify them for your own purposes

Have you learned something useful?

[I would love to know what interested you the most]
[...and the least]

Code for the examples on the Linux kernel: `https://github.com/VizGrimoire/VizGrimoireR/tree/master/examples/linux`
Databases: `http://bitergia.com/public/previews/2013_10_linux/browser/data/db`