Software Heritage

Analyzing the Global Graph of Public Software Development

Stefano Zacchiroli

Télécom Paris, Institut Polytechnique de Paris
stefano.zacchiroli@telecom-paris.fr

7 Oct 2022
ENS Rennes

Software Heritage
THE GREAT LIBRARY OF SOURCE CODE

Outline

About the speaker

- Professor of Computer Science, Télécom Paris, Institut Polytechnique de Paris
- Free/Open Source Software activist (20+ years)
- Debian Developer & Former 3x Debian Project Leader
- Former Open Source Initiative (OSI) director
- Software Heritage co-founder & CTO

Outline

Software is everywhere (and a key mediator) in society

Free/Open Source Software (FOSS) is everywhere as well

Definition (Free Software (1986))

A program is free software if the program's users have the four *essential freedoms*:

- Freedom #0, to run the program, for any purpose
- Freedom #1, to study how the program works, and change it
- Freedom #2, to redistribute copies
- Freedom #3, to improve the program, and release improvements

Current estimates: 99% of software products on the market contains at least *some* free software parts (Synopsis 2020).

The Commons and FOSS

Definition (Commons)

The commons is the cultural and natural resources accessible to all members of a society, including natural materials such as air, water, and a habitable earth. These resources are held in common, not owned privately.

Definition (Software Commons)

The software commons consists of all computer software which is available at little or no cost and which can be altered and reused with few restrictions. Thus *all open source software and all free software are part of the [software] commons.* [...]

Kranich and Schement (2008); Schweik and English (2012).

Software *source code* is precious human knowledge

Harold Abelson, Structure and Interpretation of Computer Programs (1st ed.) 1985

"Programs must be written for people to read, and only incidentally for machines to execute."

Software *source code* is precious human knowledge

Harold Abelson, Structure and Interpretation of Computer Programs (1st ed.) 1985

"Programs must be written for people to read, and only incidentally for machines to execute."

Apollo 11 source code (excerpt)

```
P63SPOT3        CA      BIT6            # IS THE LR ANTENNA IN POSITION 1 YET
                EXTEND
                RAND    CHAN33
                EXTEND
                BZF     P63SPOT4        # BRANCH IF ANTENNA ALREADY IN POSITION 1

                CAF     CODE500         # ASTRONAUT:    PLEASE CRANK THE
                TC      BANKCALL        #               SILLY THING AROUND
                CADR    GOPERF1
                TCF     GOTOPOOH        # TERMINATE
                TCF     P63SPOT3        # PROCEED       SEE IF HE'S LYING

P63SPOT4        TC      BANKCALL        # ENTER         INITIALIZE LANDING RADAR
                CADR    SETPOS1

                TC      POSTJUMP        # OFF TO SEE THE WIZARD ...
                CADR    BURNBABY
```

Software *source code* is precious human knowledge

Harold Abelson, Structure and Interpretation of Computer Programs (1st ed.) 1985

"Programs must be written for people to read, and only incidentally for machines to execute."

Apollo 11 source code (excerpt)

```
P63SPOT3        CA      BIT6            # IS THE LR ANTENNA IN POSITION 1 YET
                EXTEND
                RAND    CHAN33
                EXTEND
                BZF     P63SPOT4        # BRANCH IF ANTENNA ALREADY IN POSITION 1

                CAF     CODE500         # ASTRONAUT:    PLEASE CRANK THE
                TC      BANKCALL        #               SILLY THING AROUND
                CADR    GOPERF1
                TCF     GOTOPOOH        # TERMINATE
                TCF     P63SPOT3        # PROCEED       SEE IF HE'S LYING

P63SPOT4        TC      BANKCALL        # ENTER         INITIALIZE LANDING RADAR
                CADR    SETPOS1

                TC      POSTJUMP        # OFF TO SEE THE WIZARD ...
                CADR    BURNBABY
```

Quake III source code (excerpt)

```
float Q_rsqrt( float number )
{
    long i;
    float x2, y;
    const float threehalfs = 1.5F;

    x2 = number * 0.5F;
    y = number;
    i = * ( long * ) &y; // evil floating point bit level hacking
    i = 0x5f3759df - ( i >> 1 ); // what the fuck?
    y = * ( float * ) &i;
    y = y * ( threehalfs - ( x2 * y * y ) ); // 1st iteration
// y = y * ( threehalfs - ( x2 * y * y ) ); // 2nd iteration, this
can be removed

    return y;
}
```

Software *source code* is precious human knowledge

Harold Abelson, Structure and Interpretation of Computer Programs (1st ed.) 1985

"Programs must be written for people to read, and only incidentally for machines to execute."

Apollo 11 source code (excerpt)

```
P63SPOT3        CA      BIT6            # IS THE LR ANTENNA IN POSITION 1 YET
                EXTEND
                RAND    CHAN33
                EXTEND
                BZF     P63SPOT4        # BRANCH IF ANTENNA ALREADY IN POSITION 1

                CAF     CODE500         # ASTRONAUT:    PLEASE CRANK THE
                TC      BANKCALL        #               SILLY THING AROUND
                CADR    GOPERF1
                TCF     GOTOPOOH        # TERMINATE
                TCF     P63SPOT3        # PROCEED       SEE IF HE'S LYING

P63SPOT4        TC      BANKCALL        # ENTER         INITIALIZE LANDING RADAR
                CADR    SETPOS1

                TC      POSTJUMP        # OFF TO SEE THE WIZARD ...
                CADR    BURNBABY
```

Quake III source code (excerpt)

```
float Q_rsqrt( float number )
{
    long i;
    float x2, y;
    const float threehalfs = 1.5F;

    x2 = number * 0.5F;
    y = number;
    i = * ( long * ) &y; // evil floating point bit level hacking
    i = 0x5f3759df - ( i >> 1 ); // what the fuck?
    y = * ( float * ) &i;
    y = y * ( threehalfs - ( x2 * y * y ) ); // 1st iteration
// y = y * ( threehalfs - ( x2 * y * y ) ); // 2nd iteration, this
can be removed

    return y;
}
```

Len Shustek, Computer History Museum 2006

"Source code provides a view into the mind of the designer."

Software source code is fragile

Like all digital information, FOSS is fragile

- link rot: projects are created, moved around, removed
- business-driven code loss (e.g., Gitorious, Google Code, Bitbucket)
- data rot: physical media with legacy software decay

Software source code is fragile

Like all digital information, FOSS is fragile

- link rot: projects are created, moved around, removed
- business-driven code loss (e.g., Gitorious, Google Code, Bitbucket)
- data rot: physical media with legacy software decay

If a website disappears you go to the Internet Archive…

where do you go if (a repository on) GitHub or GitLab goes away?

Outline

Software Heritage

THE GREAT LIBRARY OF SOURCE CODE

Collect, preserve and share *all* software source code

Preserving our heritage, enabling better software and better science for all

Software Heritage

THE GREAT LIBRARY OF SOURCE CODE

Collect, preserve and share *all* software source code

Preserving our heritage, enabling better software and better science for all

Reference catalog

find and reference all software source code

Software Heritage

THE GREAT LIBRARY OF SOURCE CODE

Collect, preserve and share *all* software source code

Preserving our heritage, enabling better software and better science for all

Reference catalog

find and reference all software source code

Universal archive

preserve and share all software source code

Software Heritage

THE GREAT LIBRARY OF SOURCE CODE

Collect, preserve and share *all* software source code

Preserving our heritage, enabling better software and better science for all

Reference catalog

find and reference all software source code

Universal archive

preserve and share all software source code

Research infrastructure

enable analysis of all software source code

Archiving goals

Targets: VCS repositories & source code releases (e.g., tarballs, packages)

We DO archive

- file content (= blobs)
- revisions (= commits), with full metadata
- releases (= tags), ditto
- where (origin) & when (visit) we found any of the above

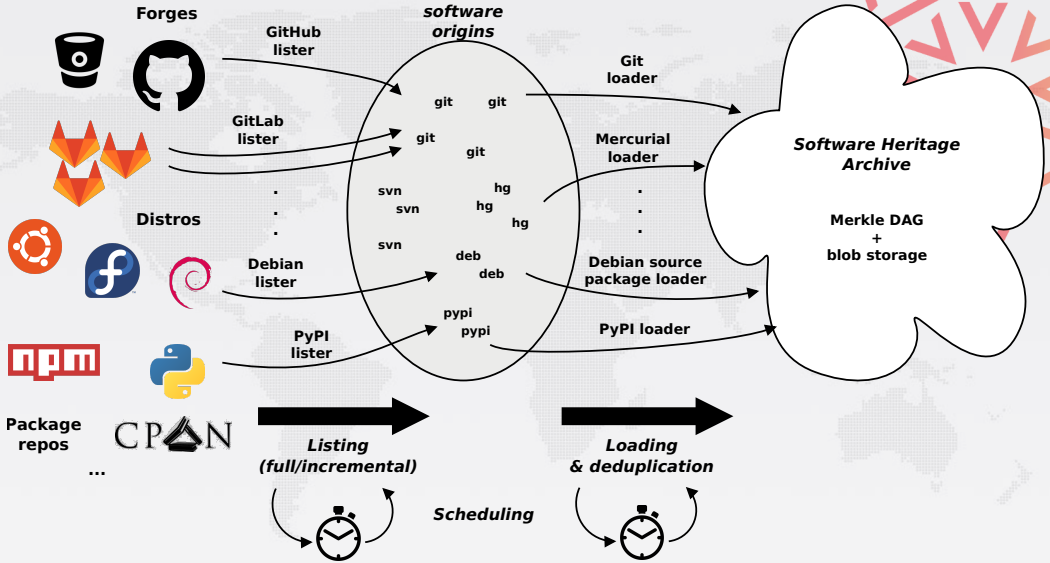
… in a VCS-/archive-agnostic canonical data model

We DON'T archive (yet)

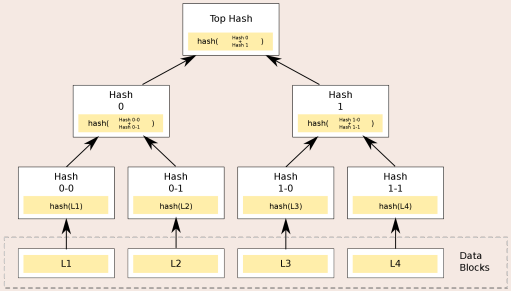
- homepages, wikis
- BTS/issues/code reviews/etc.
- mailing lists

Long term vision: play our part in a *"semantic wikipedia of software"*

Data flow


Forges
GitHub lister
GitLab lister
Distros
Debian lister
PyPI lister
Package repos
CPAN
...
software origins
git
git
git
git
svn
svn
svn
hg
hg
hg
deb
deb
pypi
pypi
Git loader
Mercurial loader
Debian source package loader
PyPI loader
Software Heritage Archive
Merkle DAG
+
blob storage
Listing (full/incremental)
Scheduling
Loading & deduplication

Merkle tree (R. C. Merkle, CRYPTO 1987)

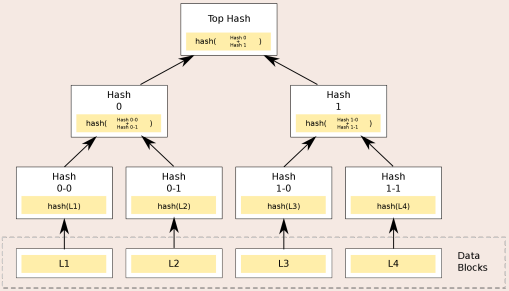


Combination of

- tree
- hash function

Merkle trees

Merkle tree (R. C. Merkle, CRYPTO 1987)



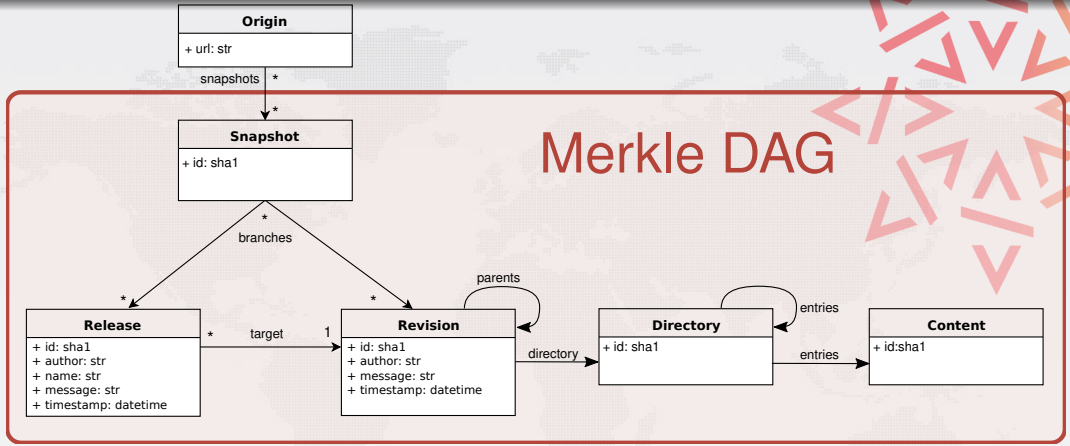
Combination of

- tree
- hash function

Classical cryptographic construction

- fast, parallel signature of large data structures
- widely used (e.g., Git, blockchains, IPFS, …)
- built-in deduplication

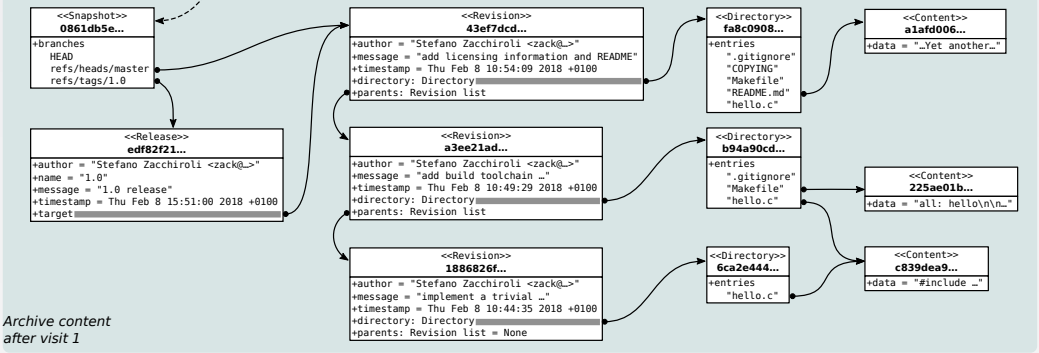
Data model



A global graph linking together fully deduplicated source code artifact (files, commits, directories, releases, etc.) to the places that distribute them (e.g., Git repositories), providing a unified view on the entire *Software Commons*.

The archive: a (giant) Merkle DAG

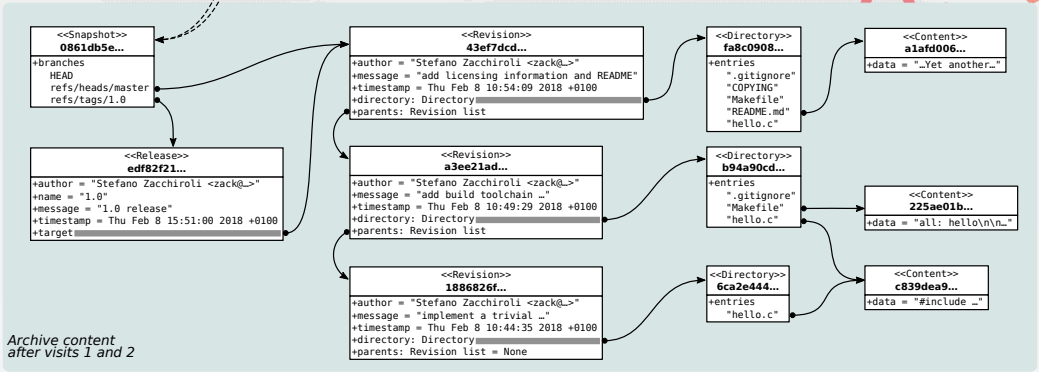
origin	visit	snapshot	timestamp
https://forge.softwareheritage.org/source/helloworld.git	1	0861db5e…	Fri Feb 9 12:38:45 2018 +0100



Archive content after visit 1

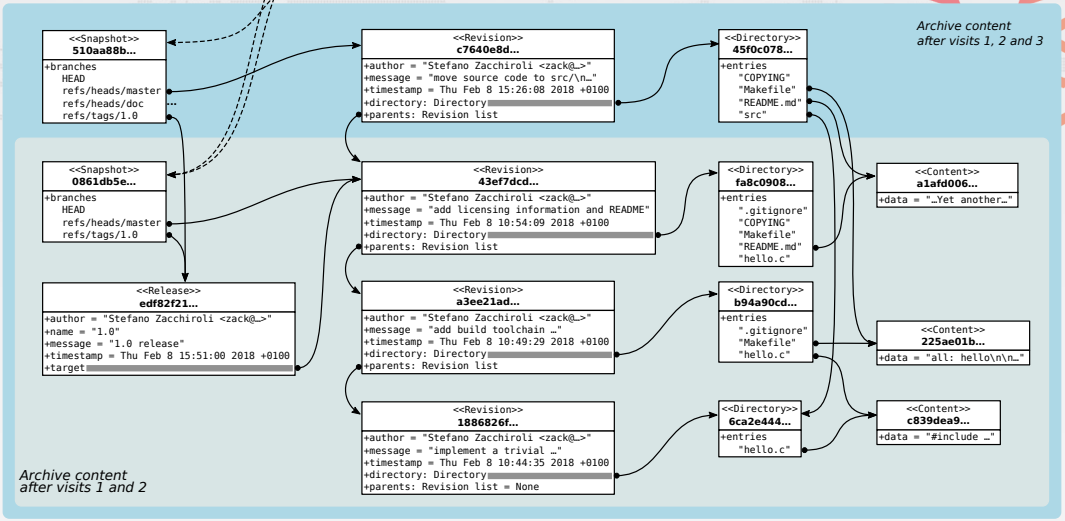
The archive: a (giant) Merkle DAG

origin	visit	snapshot	timestamp
https://forge.softwareheritage.org/source/helloworld.git	1	0861db5e...	Fri Feb 9 12:38:45 2018 +0100
https://forge.softwareheritage.org/source/helloworld.git	2	0861db5e...	Fri Feb 9 13:29:00 2018 +0100



The archive: a (giant) Merkle DAG

origin	visit	snapshot	timestamp
https://forge.softwareheritage.org/source/helloworld.git	1	0861db5e…	Fri Feb 9 12:38:45 2018 +0100
https://forge.softwareheritage.org/source/helloworld.git	2	0861db5e…	Fri Feb 9 13:29:00 2018 +0100
https://forge.softwareheritage.org/source/helloworld.git	3	510aa88b…	Fri Feb 9 15:52:50 2018 +0100



Archive coverage — archive.softwareheritage.org

Bitbucket · R · debian · Framagit · GitHub · GitLab · GITORIOUS · Google code · {P} · GNU · Guix · HAL archives-ouvertes.fr · Inria · IPOL Journal · npm · NixOS · python

Archive coverage — archive.softwareheritage.org

Bitbucket · R · debian · Framagit · GitHub · GitLab · GITORIOUS · Google code · GNU · Guix · HAL archives-ouvertes.fr · Inria · IPOL Journal · npm · NixOS · python

- on disk: ~1 PiB; as a graph ~25 B nodes, ~350 B edges
- the largest public source code archive in the world (and growing!)

Demo time!

- Browse the archive
- Trigger archival of your preferred software in a breeze
- Get and use SWHIDs (full specification available online)
- The Apollo 11 AGC source code example
- Cite software with the biblatex-software style from CTAN
- Example use in a research article: compare Fig. 1 and conclusions
  - in the 2012 version
  - in the updated version using SWHIDs and Software Heritage
- Example in a journal: an article from IPOL
- Curated deposit in SWH via HAL, see for example: LinBox, SLALOM, Givaro, NS2DDV, SumGra, Coq proof, …
- Rescue landmark legacy software, see the SWHAP process with UNESCO

Outline

Graph dataset

Use case: large scale analyses of the most comprehensive corpus on the development history of free/open source software.

Antoine Pietri, Diomidis Spinellis, Stefano Zacchiroli
The Software Heritage Graph Dataset: Public software development under one roof
MSR 2019: 16th Intl. Conf. on Mining Software Repositories. IEEE
preprint: http://deb.li/swhmsr19

Dataset

- Relational representation of the full graph as a set of tables
- Available as open data: https://doi.org/10.5281/zenodo.2583978
- Chosen as subject for the MSR 2020 Mining Challenge

Formats

- Local use: PostgreSQL dumps, or Apache Parquet files (~1 TiB each)
- Live usage: Amazon Athena (SQL-queriable), Azure Data Lake

Graph dataset — example

```
SELECT COUNT(*) AS c, word FROM (
  SELECT LOWER(REGEXP_EXTRACT(FROM_UTF8(
  message), '^\w+')) AS word FROM revision)
WHERE word != ''
GROUP BY word ORDER BY COUNT(*) DESC LIMIT 5;
```

Graph dataset — example

```
SELECT COUNT(*) AS c, word FROM (
  SELECT LOWER(REGEXP_EXTRACT(FROM_UTF8(
  message), '^\w+')) AS word FROM revision)
WHERE word != ''
GROUP BY word ORDER BY COUNT(*) DESC LIMIT 5;
```

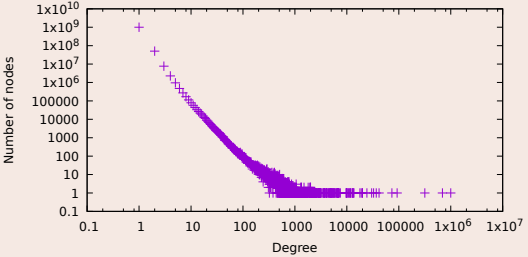
Count	Word
71 338 310	update
64 980 346	merge
56 854 372	add
44 971 954	added
33 222 056	fix

Graph dataset — example

Fork arity

i.e., how often is a commit based upon?

```
SELECT fork_deg, count(*) FROM (
  SELECT id, count(*) AS fork_deg
  FROM revision_history GROUP BY id) t
GROUP BY fork_deg ORDER BY fork_deg;
```

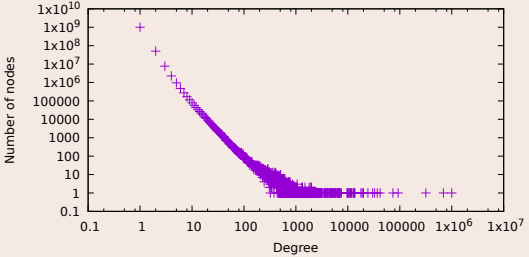


Graph dataset — example

Fork arity

i.e., how often is a commit based upon?

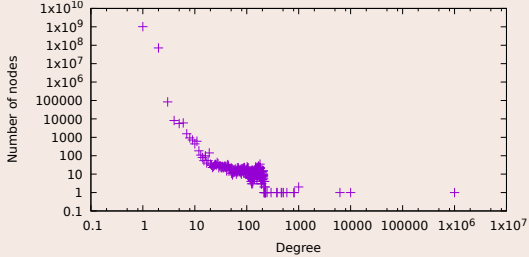
```
SELECT fork_deg, count(*) FROM (
  SELECT id, count(*) AS fork_deg
  FROM revision_history GROUP BY id) t
GROUP BY fork_deg ORDER BY fork_deg;
```



Merge arity

i.e., how large are merges?

```
SELECT merge_deg, COUNT(*) FROM (
  SELECT parent_id, COUNT(*) AS merge_deg
  FROM revision_history GROUP BY parent_id)
GROUP BY merge_deg ORDER BY merge_deg;
```



License dataset

Stefano Zacchiroli
A Large-scale Dataset of (Open Source) License Text Variants
MSR 2022 (best dataset paper award)
preprint: https://arxiv.org/abs/2204.00256

Dataset

- 6.5 million unique full texts of FOSS license variants
- Detected using filename patterns across the entire SWH archive
  - LICENSE, COPYRIGHT, NOTICE, etc.
- Metadata: file lengths measures, detected MIME type, detected SPDX license (via ScanCode), example origin repository, oldest public commit of origin

Use cases

- Empirical studies on FOSS licensing, including phylogenetics
- Training of automated license classifiers
- NLP analyses of legal texts

The Software Heritage Filesystem (SwhFS)

The Software Heritage Filesystem (SwhFS) is a user-space POSIX filesystem that enables browsing parts of the Software Heritage archive as if it were locally available.

- code: forge.softwareheritage.org/source/swh-fuse
- documentation: docs.softwareheritage.org/devel/swh-fuse

Thibault Allançon, Antoine Pietri, Stefano Zacchiroli
The Software Heritage Filesystem (SwhFS): Integrating Source Code Archival with Development
ICSE 2021: The 43rd International Conference on Software Engineering
https://arxiv.org/abs/2102.06390

The Software Heritage Filesystem (SwhFS) — example

```
$ mkdir swhfs
$ swh fs mount swhfs/  # mount the archive
$ cd swhfs/

$ cat archive/swh:1:cnt:c839dea9e8e6f0528b468214348fee8669b305b2
#include <stdio.h>

int main(void) {
    printf("Hello, World!\n");
}

$ cd archive/swh:1:dir:1fee702c7e6d14395bbf5ac3598e73bcbf97b030
$ ls | wc -l
127
$ grep -i antenna THE_LUNAR_LANDING.s | cut -f 5
# IS THE LR ANTENNA IN POSITION 1 YET
# BRANCH IF ANTENNA ALREADY IN POSITION 1
```

The Software Heritage Filesystem (SwhFS) — example (cont.)

```
$ cd archive/swh:1:rev:9d76c0b163675505d1a901e5fe5249a2c55609bc

$ ls -F
history/  meta.json@  parent@  parents/  root@

$ jq '.author.name, .date, .message' meta.json
"Michal Golebiowski-Owczarek"
"2020-03-02T23:02:42+01:00"
"Data:Event:Manipulation: Prevent collisions with Object.prototype ..."

$ find root/src/ -type f -name '*.js' | xargs cat | wc -l
10136
```

Graph compression

Paolo Boldi, Antoine Pietri, Sebastiano Vigna, Stefano Zacchiroli

Ultra-Large-Scale Repository Analysis via Graph Compression

SANER 2020, 27th Intl. Conf. on Software Analysis, Evolution and Reengineering. IEEE

Research question

Is it possible to efficiently perform software development history analyses at the scale of Software Heritage archive on a single, relatively cheap machine?

Idea

Apply state-of-the-art graph compression techniques from the field of Web graph / social network analysis.

Results

The entire archive graph (25 B nodes, 350 B edges) can be loaded in 200 GiB and then traversed at the cost of tens of ns per edge (= a few hours for a full single-thread visit).

Java and gRPC APIs available: docs.softwareheritage.org/devel/swh-graph/grpc-api.html

Background — (Web) graph compression

Definition (The graph of the Web)

Directed graph that has Web pages as nodes and hyperlinks between them as edges.

Properties (1)

- Locality: pages link to pages whose URLs are lexicographically similar. URLs share long common prefixes.

→ use D-gap compression

Adjacency lists

Node	Outdegree	Successors
…	…	…
15	11	13,15,16,17,18,19,23,24,203,315,1034
16	10	15,16,17,22,23,24,315,316,317,3041
17	0	
18	5	13,15,16,17,50
…	…	…

D-gapped adjacency lists

Node	Outdegree	Successors
…	…	…
15	11	3,1,0,0,0,0,3,0,178,111,718
16	10	1,0,0,4,0,0,290,0,0,2723
17	0	
18	5	9,1,0,0,32
…	…	…

Background — (Web) graph compression (cont.)

Definition (The graph of the Web)

Directed graph that has Web pages as nodes and hyperlinks between them as edges.

Properties (2)

- Similarity: pages that are close together in lexicographic order tend to have many common successors.

→ use reference compression

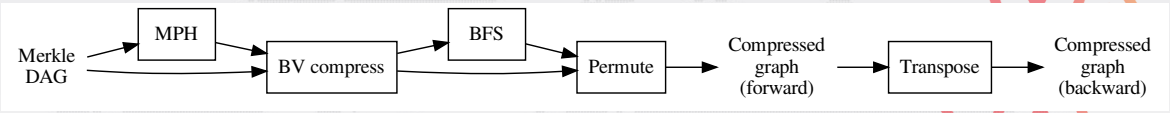
Adjacency lists

Node	Outd.	Successors
…	…	…
15	11	13,15,16,17,18,19,23,24,203,315,1034
16	10	15,16,17,22,23,24,315,316,317,3041
17	0	
18	5	13,15,16,17,50
…	…	…

Copy lists

Node	Ref.	Copy list	Extra nodes
…	…	…	…
15	0		13,15,16,17,18,19,23,24,203,315,1034
16	1	01110011010	22,316,317,3041
17			
18	3	11110000000	50
…	…	…	

Graph compression pipeline

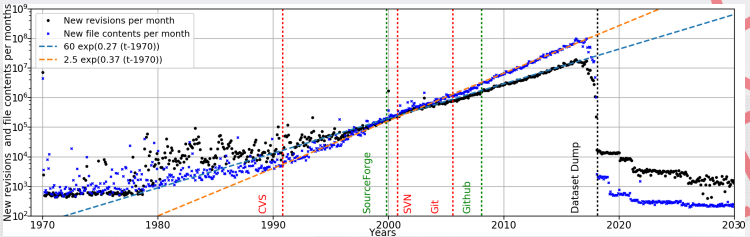


- MPH: minimal perfect hash, mapping Merkle IDs to 0..N-1 integers
- BV compress: Boldi-Vigna compression (based on MPH order)
- BFS: breadth-first visit to renumber
- Permute: update BV compression according to BFS order

(Re)establishing locality

- key for good compression is a node ordering that ensures locality and similarity
- which is very much *not* the case with Merkle IDs, … but is the case *again* after BFS reordering

Software provenance and evolution



Key findings

- The amount of original commits in public code doubles every ~30 months and has been doing so for 20+ years; original source code files double every ~22 months
- It is possible to trace the provenance of source code artifacts at this scale in a compact relational model via the notion of isochrone graphs.

Rousseau, Di Cosmo, Zacchiroli

Software Provenance Tracking at the Scale of Public Source Code

In Empirical Software Engineering, 2020

Diversity, equity, and inclusion

Idea

Archived commit metadata contains public information that can be mined to study long-term trends of diversity, equity, and inclusion (DEI) traits of the global population of public code contributors.

Key findings on the gender gap

- Male authors contributed 92% of public code commits up to 2019.
- The ratio of female authors (and their contributions) has grown stably for 15 years reaching for the first time 10% of yearly contributions in 2019.
- The COVID-19 pandemic has reversed the trend.

Diversity, equity, and inclusion (cont.)

Key findings on the geographic gap

- The early decades of public code were dominated by contributions from North America, followed by a period of alternating dominance between North America and Europe.
- Since then geographic diversity has increased constantly, with raising importance of contributions from Central and South America.
- The trend of increased female contributions is almost worlwide, with the notable exception of specific regions of Asia were it is either slower or flat.

References

- Zacchiroli. *Gender differences in public code contributions: a 50-year perspective.* IEEE Software, 2021
- Rossi and Zacchiroli. *Worldwide gender differences in public code contributions (and how they have been affected by the COVID-19 pandemic).* ICSE SEIS, 2022
- Rossi and Zacchiroli. *Geographic diversity in public code contributions.* MSR 2022

Outline

Conclusion

- Software Heritage archives public code and its history as a huge Merkle DAG
- Querying and analyzing it at scale (25/350 B nodes/edges) is an open problem
- Gold mine of research leads in and around empirical software engineering

Coding

- Developer info: www.softwareheritage.org/community/developers

Work with us

- Open positions (tech & research): www.softwareheritage.org/jobs

Student opportunities

- Internships, GSoC/Outreachy, student grants
- www.softwareheritage.org/community/students

Appendix

Compression experiment

Step	Wall time (hours)
MPH	2
BV Compress	84
BFS	19
Permute	18
Transpose	15
Total	138 (6 days)

- server equipped with 24 CPUs and 750 GB of RAM
- RAM mostly used as I/O cache for the BFS step
- *minimum* memory requirements are close to the RAM needed to load the final compressed graph in memory

Compression efficiency (space)

Forward graph	
total size	91 GiB
bits per edge	4.91
compression ratio	15.8%

Backward graph	
total size	83 GiB
bits per edge	4.49
compression ratio	14.4%

Operating cost

The structure of a full bidirectional archive graph fits in less than 200 GiB of RAM, for a hardware cost of ~300 USD.

Compression efficiency (time)

Benchmark — Full BFS visit (single thread)

Forward graph		Backward graph	
wall time	1h48m	wall time	3h17m
throughput	1.81 M nodes/s (553 ns/node)	throughput	988 M nodes/s (1.01 µs/node)

Benchmark — Edge lookup

random sample: 1 B nodes (8.3% of entire graph); then enumeration of all successors

Forward graph		Backward graph	
visited edges	13.6 B	visited edges	13.6 B
throughput	12.0 M edges/s (83 ns/edge)	throughput	9.45 M edges/s (106 ns/edge)

Note how edge lookup time is close to DRAM random access time (50-60 ns).

Discussion

Incrementality

compression is not incremental, due to the use of contiguous integer ranges

- but the graph is append-only, so…
- …based on expected graph growth rate it should be possible to pre-allocate enough free space in the integer ranges to support amortized incrementality (future work)

Discussion

Incrementality

compression is not incremental, due to the use of contiguous integer ranges

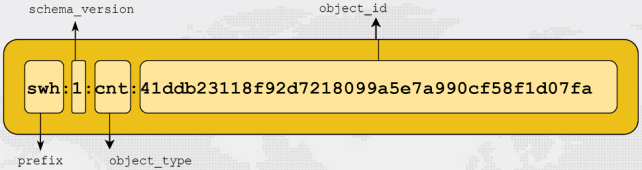
- but the graph is append-only, so…
- …based on expected graph growth rate it should be possible to pre-allocate enough free space in the integer ranges to support amortized incrementality (future work)

In-memory v. on-disk

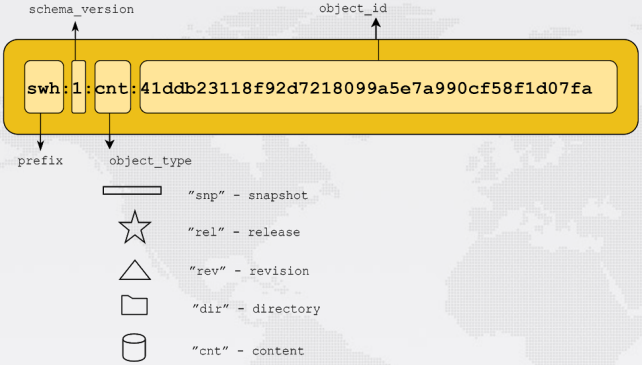
the compressed in-memory graph structure has no attributes

- usual design is to exploit the 0..N-1 integer ranges to memory map node attributes to disk for efficient access
- works well for queries that does graph traversal first and "join" node attributes last; ping-pong between the two is expensive
- edge attributes are more problematic (work in progress)

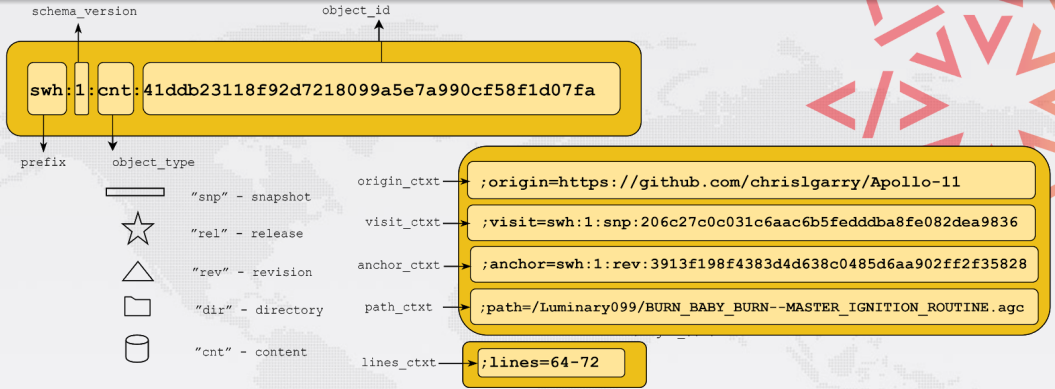
Software Heritage Identifiers (SWHIDs) (full spec)



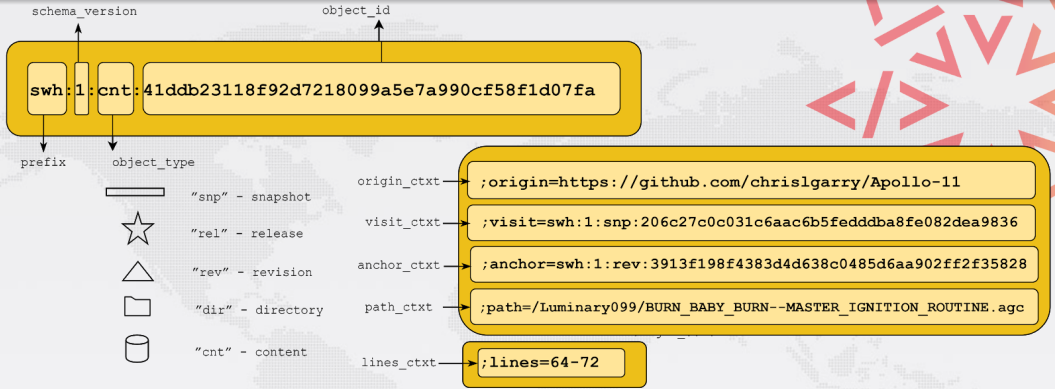
Software Heritage Identifiers (SWHIDs) (full spec)



Software Heritage Identifiers (SWHIDs) (full spec)



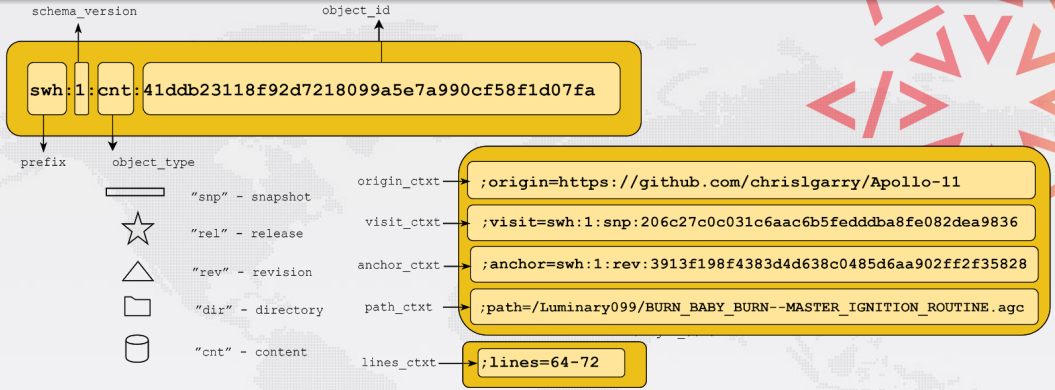
Software Heritage Identifiers (SWHIDs) (full spec)



An emerging standard

- in Linux Foundation's SPDX 2.2
- IANA-registered "swh:" URI prefix
- WikiData property P6138

Software Heritage Identifiers (SWHIDs) (full spec)



An emerging standard

- in Linux Foundation's SPDX 2.2
- IANA-registered "swh:" URI prefix
- WikiData property P6138

Examples

- Apollo 11 AGC excerpt
- Quake III rsqrt

An international, non profit initiative built for the long term

Sharing the vision

United Nations Educational, Scientific and Cultural Organization

www.softwareheritage.org/support/testimonials

Donors, members, sponsors

Diamond sponsor

Platinum sponsors

Gold sponsors

Silver sponsors

Bronze sponsors

www.softwareheritage.org/support/sponsors