Amassing and indexing a large sample of version control systems: towards the census of public source code history

Audris Mockus

audris@avaya.com

Avaya Labs Research

Basking Ridge, NJ 07920

http://mockus.org/

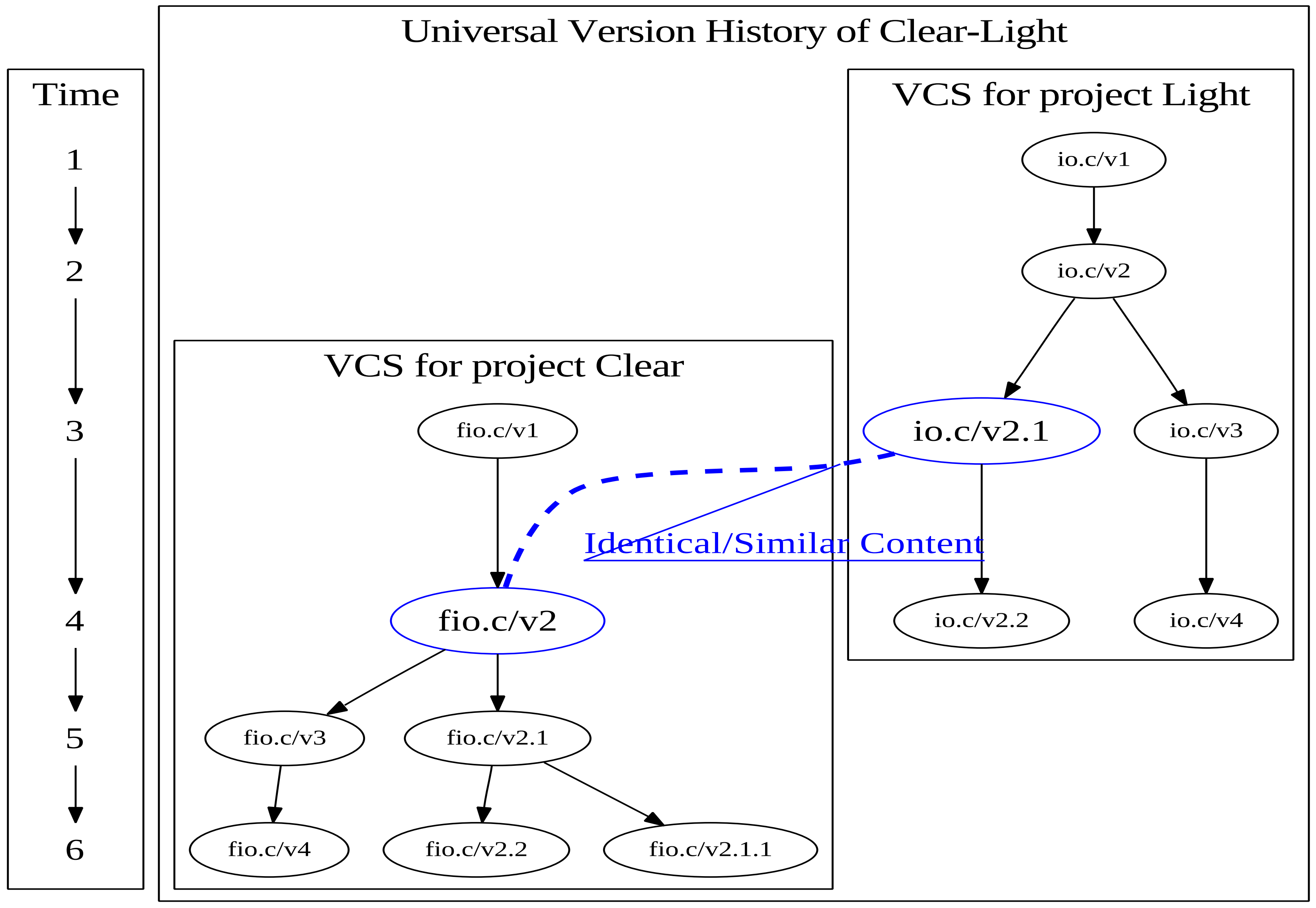
Why global properties of code?

- How much code? What is that code? How old, of what type, where?
  - Extent of code transfer/reuse: study patterns or reuse and innovation
    - Full sample needed to avoid missing instances of reuse
  - Authorship (succession): Find Adam&Eve of code or identify original authors
    - Full sample needed to avoid missing first creators
  - License compliance: verify that code is not borrowed from public domain
    - Full sample needed to avoid missing instances of borrowing

Approach: Version Control Census

- Discover VCS repositories
- Copy/clone repositories
- Establish similarity among files to determine identity of each file
  - Unlike people, files and their version histories can be and very often are copied
  - To avoid double-count for census and other analysis we thus need to create each file’s “passport” or provenance
- Conduct further analysis

Identity/provenance of the code



How to construct Universal Version History?

- Establish links among files across multiple VCS (>200M file/versions)
  - identical content: the closure of files sharing at least one identical version
  - Also: identical AST, Trigram, other ways to establish identity or similarity
- Use file/version content (AST/Trigram) as index
- Store in BerkeleyDB hashtables

Discovery strategy

- Sites with many projects: e.g., SourceForge, GoogleCode, Savannah, repo.or.cz, github.com
- Ecosystems: e.g., Gnome, KDE, NetBeans, Mozilla, ...
- Famous: e.g., Mysql, Perl, Wine, Postgres, and gcc
- In wide use: e.g., git.debian.org
- Directories: e.g., RawMeat and FSF
- Published surveys of projects
- **Verify:** search for common filenames on Google Code Search to see if new files are discovered

How to automate VCS discovery?

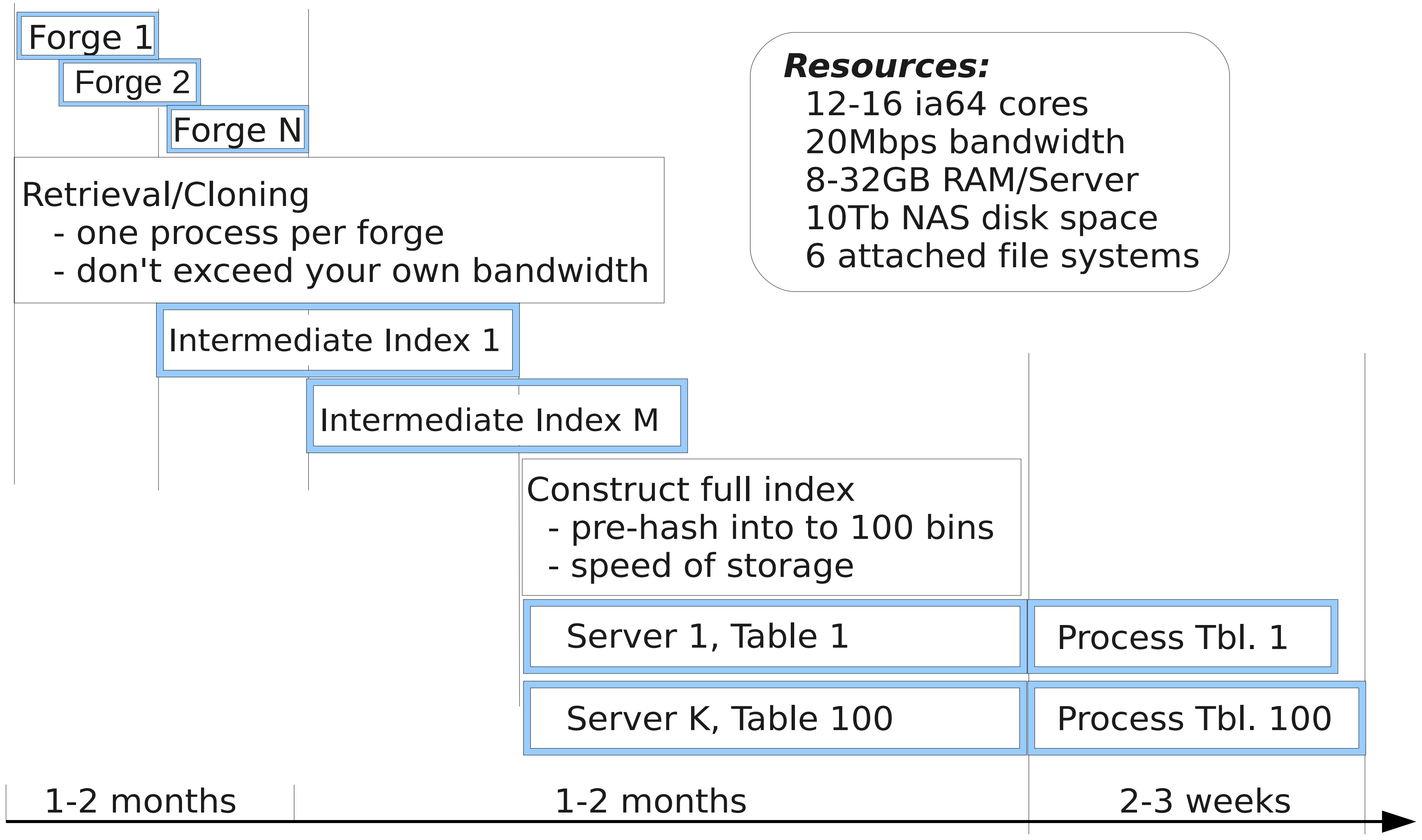
- Create a spider utilizing a search engine, and seeded by project directories (RawMeat, FSF) to grab these URLs from projects' home page
  - Search for VCS-specific URL patterns
    - cvs[:.], svn[:.], git[:.], hg[:.], bzr[:.]
- Entice projects themselves to submit a pointer to their VCS by providing a compelling service (licensing, origin, quality)
- Example discovery/update challenge
  - gitorious.org went from 68 web pages listing projects in Jan 4, 2009 to 98 last week and changed the home page format

Copy, log, extract

	URL pattern	Clone repository	List revisions
CVS	d:pserver:user@cvs.repo.org:/	rsync	cvs log
Subversion	{svn,http}://PRJ.repo.org/	svm sync URL	svn log -v URL
Git	git://git.repo.org/	git clone URL PRJ	git log OPTIONS
Mercurial	hg://hg.repo.org/	hg clone URL	hg log -v
Bazaar	http://bzr.repo.org/	bzr branch URL	

	Extract content
CVS	rcs -pREV FILE
Subversion	svn cat -rREV URL/FILE@REV
Git	git show REV:FILE
Mercurial	hg cat -rREV FILE
Bazaar	bzr cat -rREV FILE

Job scheduling: Gantt Chart



Resources:
12-16 ia64 cores
20Mbps bandwidth
8-32GB RAM/Server
10Tb NAS disk space
6 attached file systems

What is there?

Forge	Type	Files	File/Ver.	Unique File/Ver.	Branching	From
Large cmpny.	Var.	3,272K	12,585K	4,293K	2.9	1988
SourceForge	CVS	**26,095K**	81,239K	**39,550K**	2.1	1998
code.google	SVN	5,675K	14,368K	8,584K	1.7	1996
repo.or.cz	Git	2,519K	11,068K	5,115K	2.2	1986
Savannah	CVS	852K	3,623K	2,345K	1.5	**1985**
git.kernel.org	Git	12,974K	**97,585K**	856K	**114**	1988
OpenSolaris	Hg	77K	1,108K	91K	12.2	2003
FreeBSD	CVS	196K	360K	75K	4.8	1993
Kde	SVN	2,645K	10,162K	527K	19.3	1997
gnome.org	SVN	1,284K	3,981K	1,412K	2.8	1997
Gcc	SVN	3,758K	4,803K	395K	12.2	1989
Eclipse	CVS	729K	2,127K	575K	3.7	2001
OpenJDK	Hg	32K	747K	60K	12.4	2008

Implications

- Census is possible with just 4 servers
- Discovery/Update challenges
  - Brute force — a better spider
  - Carrot — compelling applications for projects to register
- VCS challenges — move to Git! (though still has no decent GUI)
  - Add a function to extract all content (version-by-version too slow)
  - Add and use author (in addition to commiter) field
  - Identify all parents of a change
- What services to provide?
  - Too big to copy — process in place
  - Start with: code origin, quality, reuse