Speeding up development time

Docker and ZFS

Mentor: Aleksandar Tošić
Matic Adamič
26.06.2023

Content

- Fast introduction to the development process
- The why and what of the system
- Introduction to Docker
- Introduction to ZFS
- Combining the two
- Research results
- Conclusion

Introduction to the development process

(from the perspective of a developer)

- You get a task (bug report, new feature, refactoring…)
- Implement with version control system (Git)
- Write a new test for it
- Run the entire test suite
- Commit to the master branch
- Eventually deploy to production

The *What and Why*

- Testing complex batch data processing programs
- Tests can take up to an hour and a half
- We have to test our feature on different databases
- Business logic can be database dependent

- What if a developer could:
  - Create an instance of a database of our choosing
  - Control start-up and shutdown of instances
  - Track changes similarly to how Git does it
- Then:
  - Developers get isolated databases
  - They can test their changes whenever and however they want
  - They can “destroy” databases beyond repair

Docker

- Engine for running light-weight VMs, known as containers (not really)
- Containers: OS-level virtualized packets
- Each container has its own separate environment, resources and file system

- Helps us isolate programs
- Helps us manage dependencies
- Helps us dynamically create or destroy instances of our programs

ZFS

- Zetta-byte file system
- “The last word in file systems”
- Lots of features:
  - Copy-on-write
  - Snapshots
  - …

ZFS: Copy-on-write

- Immutable datasets
- Dataset is a set of pointers, pointing to a block on your hard drive

- Want to modify?
  - Copy the pointers as mutable dataset
  - Do the modification

ZFS: Snapshots

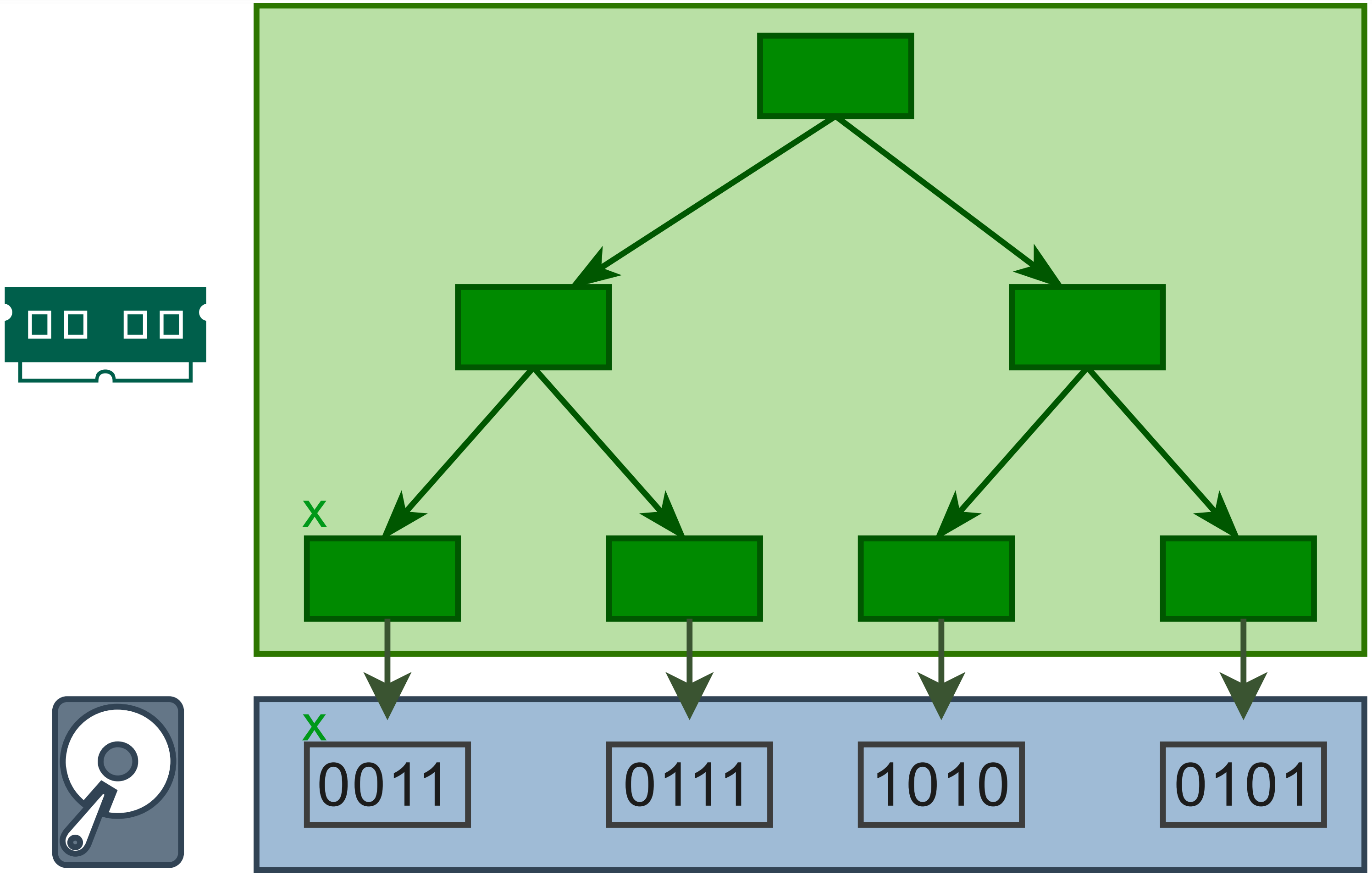
- A snapshot represents a point in time of a dataset
- Each point in time is an accessible dataset

- Snapshotting a dataset makes it immutable

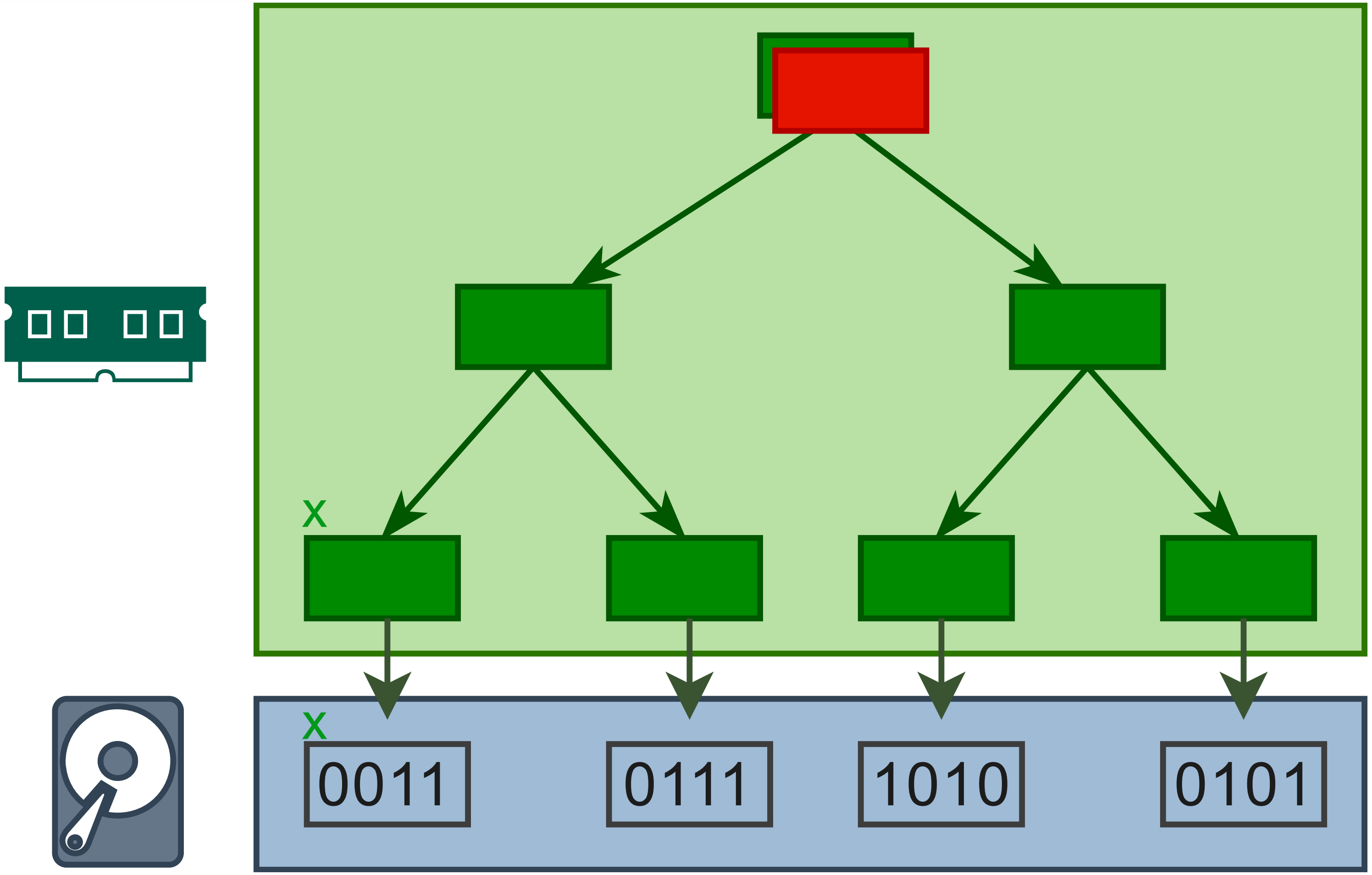
- There is branching
- There can be multiple mutable datasets of the same immutable dataset

1. Pick a snapshot of a file
2. Create a mutable dataset from it
3. Make some changes
4. Snapshot your that dataset

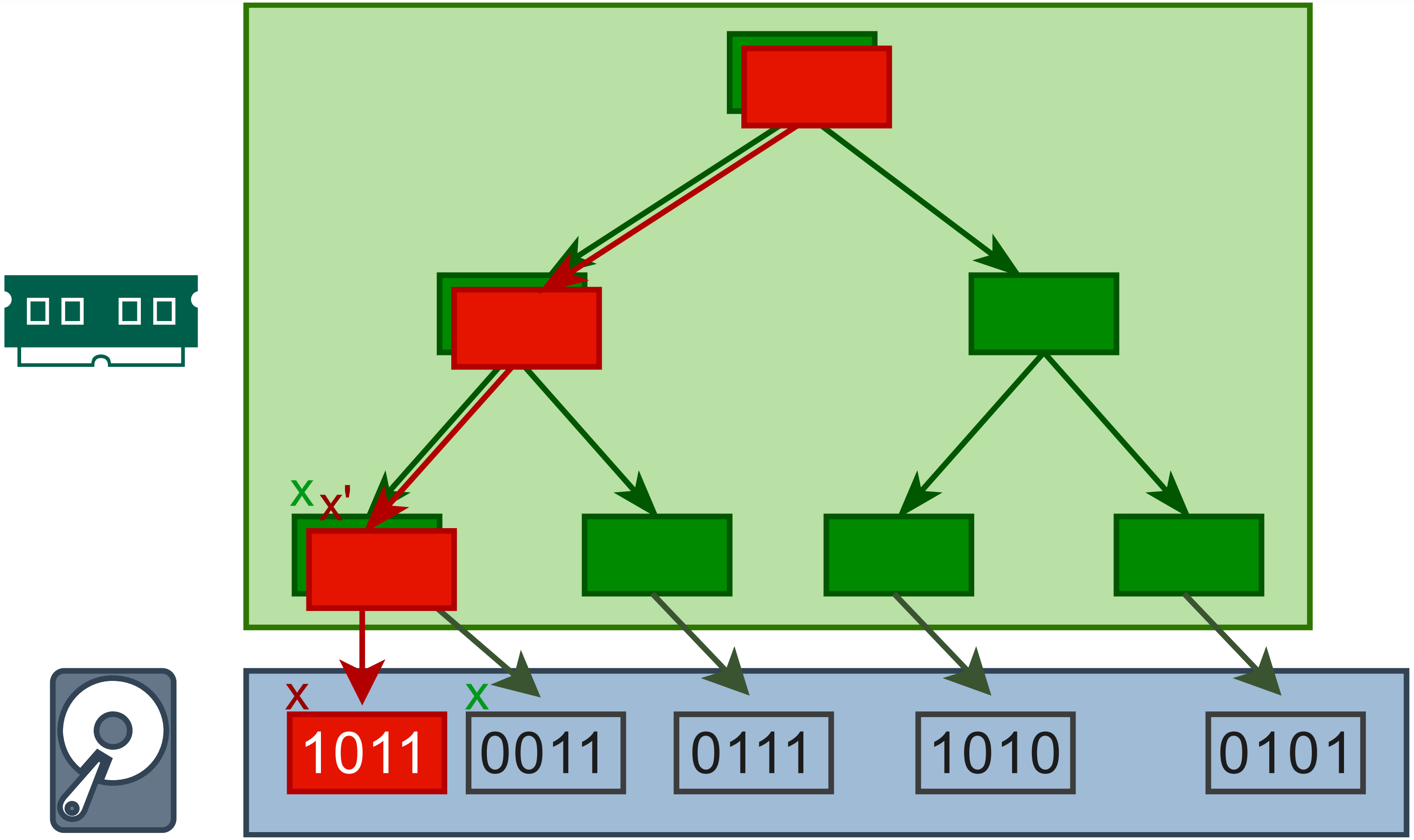
ZFS: An example



ZFS: An example

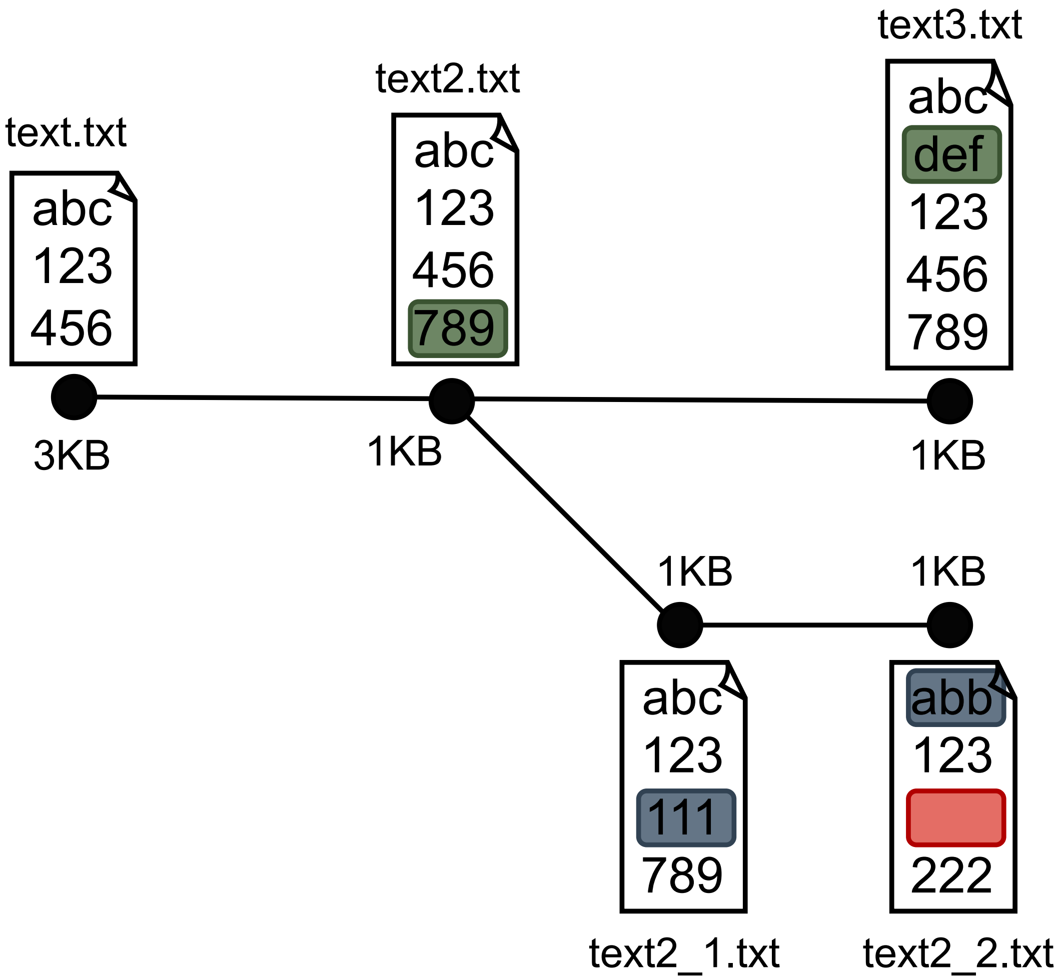


ZFS: An example



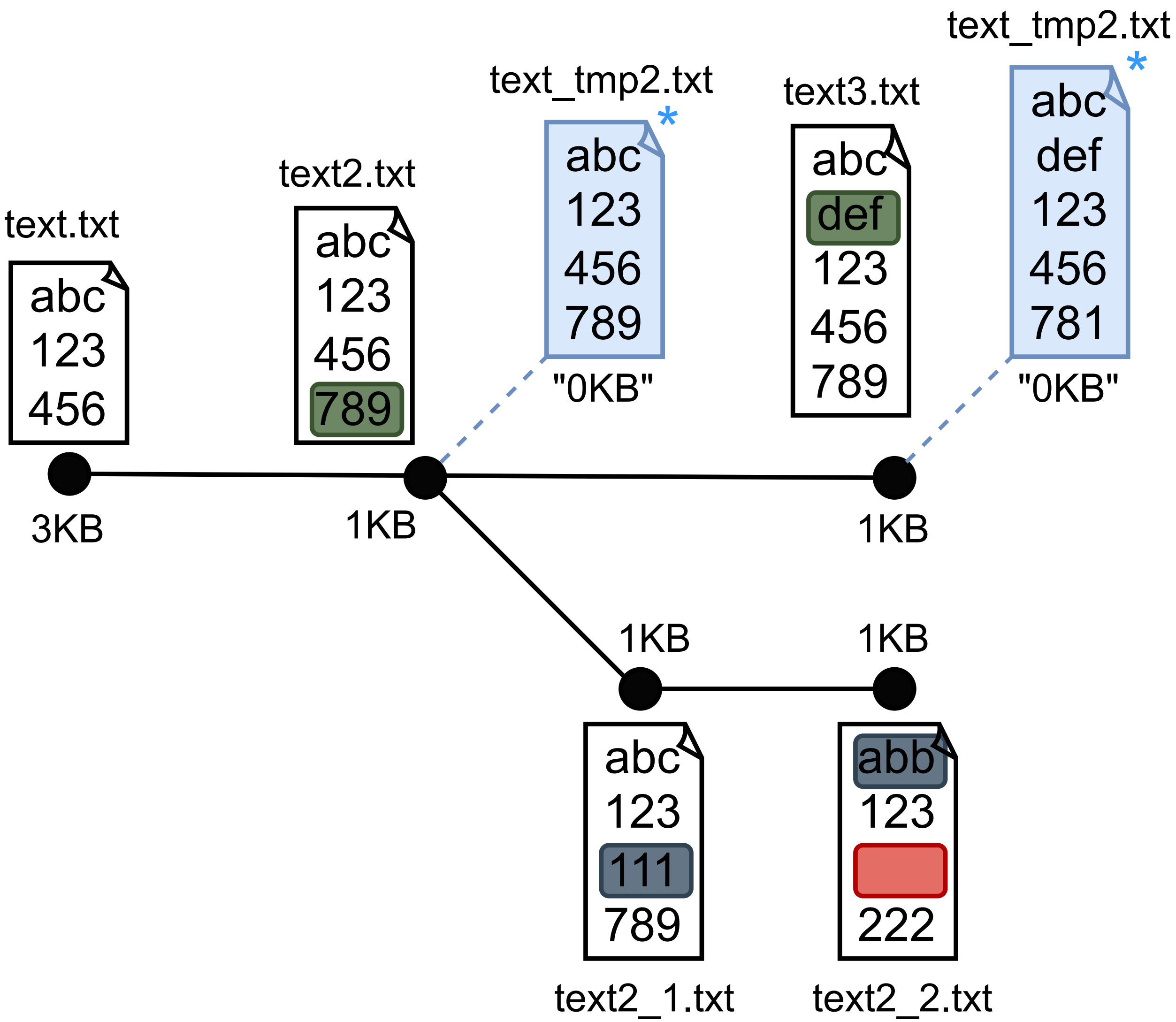
ZFS: In practice

- File history is a tree
- Changes are accumulated
- 5 files, total of 19 KB
- But only 7KB in ZFS!



ZFS: In practice

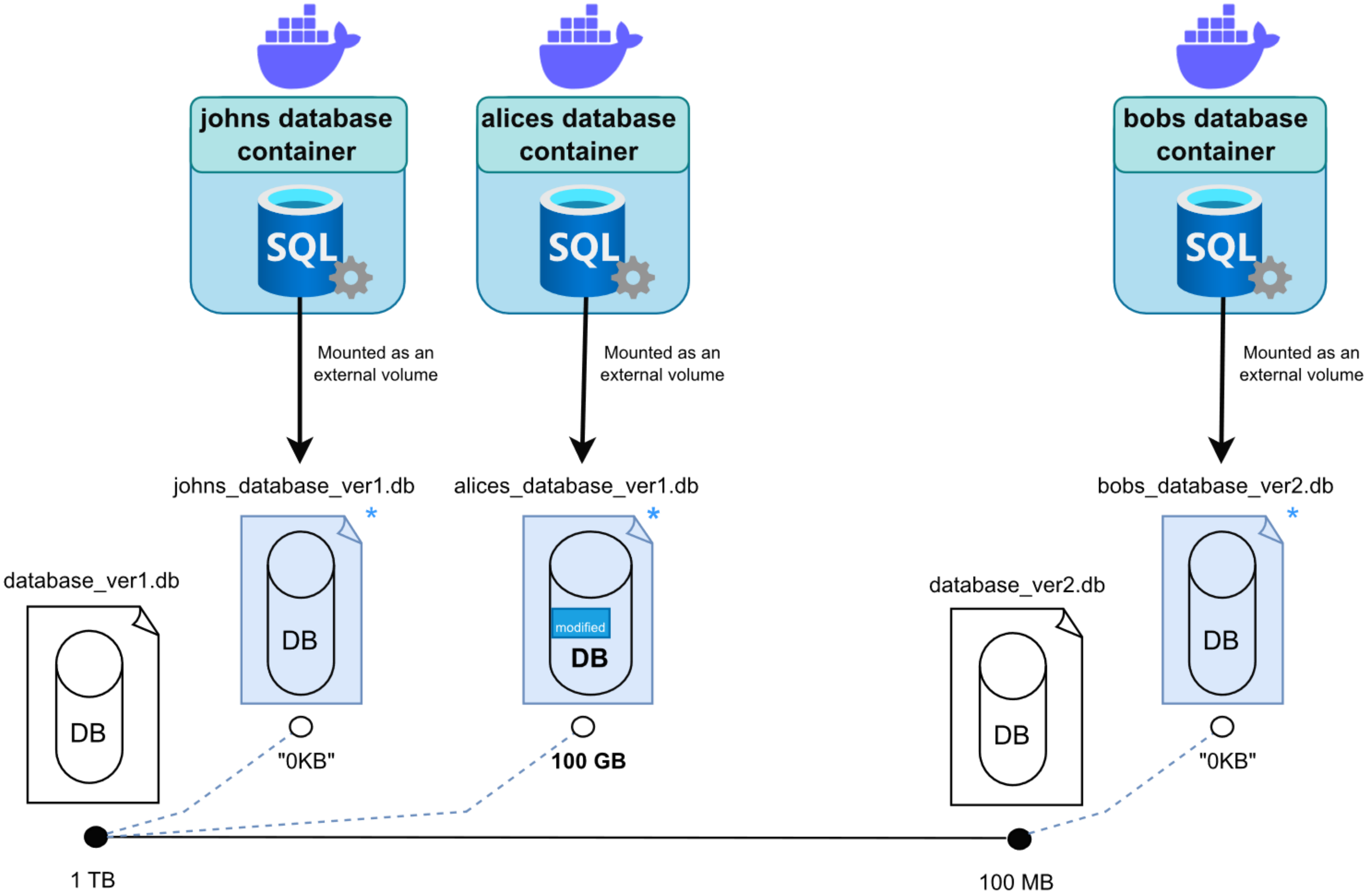
- File history is a tree
- Changes are accumulated
- 5 files, 3 KB each = 15 KB
- But only 8KB in ZFS!
- * are mutable datasets, not yet part of history (not snapshotted)



Combining ZFS and Docker

- Our dataset will be a database (a single file on your disk)
- We can mount a volume (a directory from hosts’ perspective)
- Our container will run an SQL Server inside

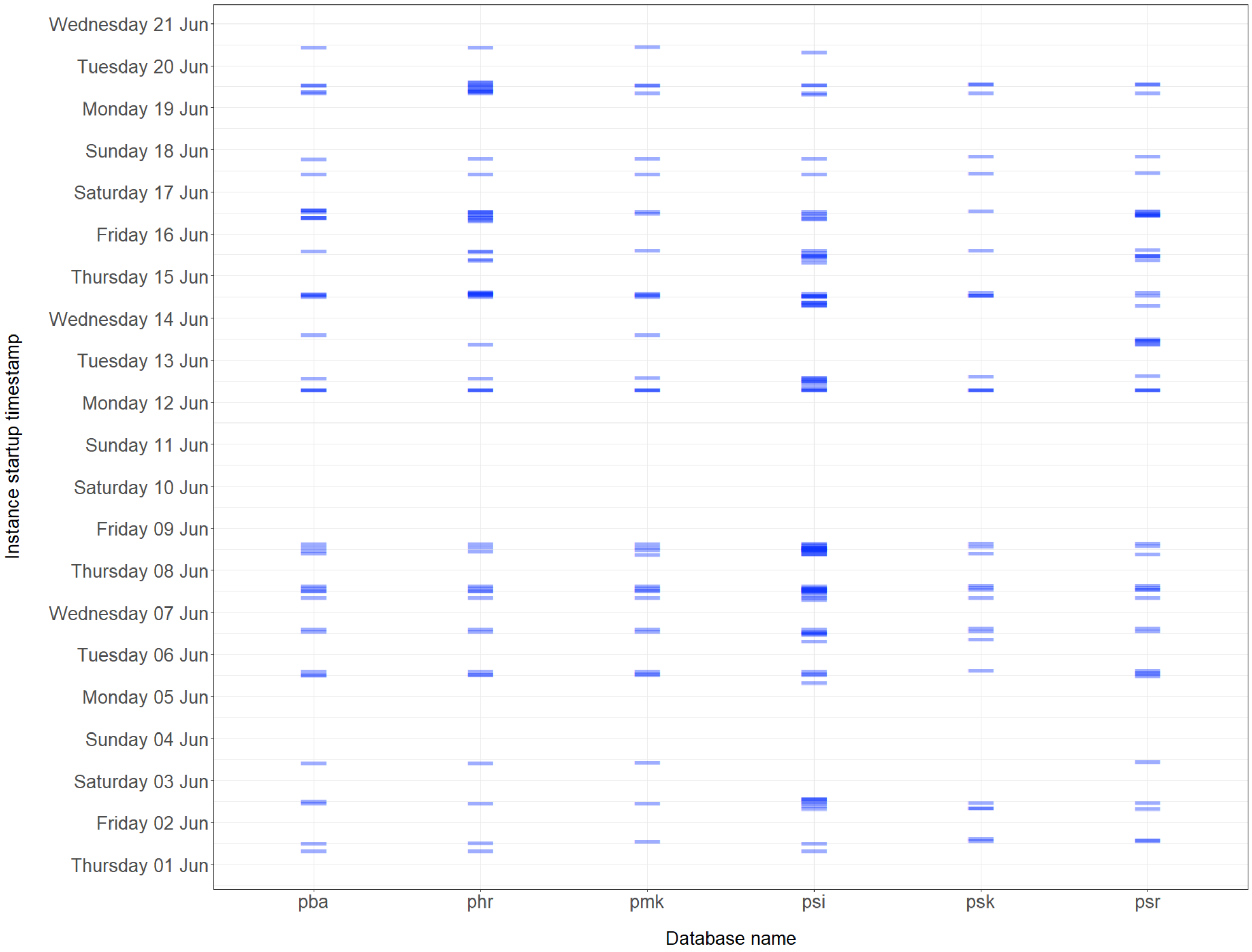
1. Turn our database file into a ZFS dataset (immutable)
2. Create a mutable dataset from it
3. Run a docker container with database dataset as a mounted volume
4. Run the SQL Server inside the container
5. Instruct SQL Server to look for a database in the mounted volume
6. Connect to database


johns database container
alices database container
bobs database container
SQL
SQL
SQL
Mounted as an external volume
Mounted as an external volume
Mounted as an external volume
johns_database_ver1.db
alices_database_ver1.db
bobs_database_ver2.db
*
*
*
database_ver1.db
database_ver2.db
DB
DB
modified
DB
DB
DB
"0KB"
100 GB
"0KB"
1 TB
100 MB

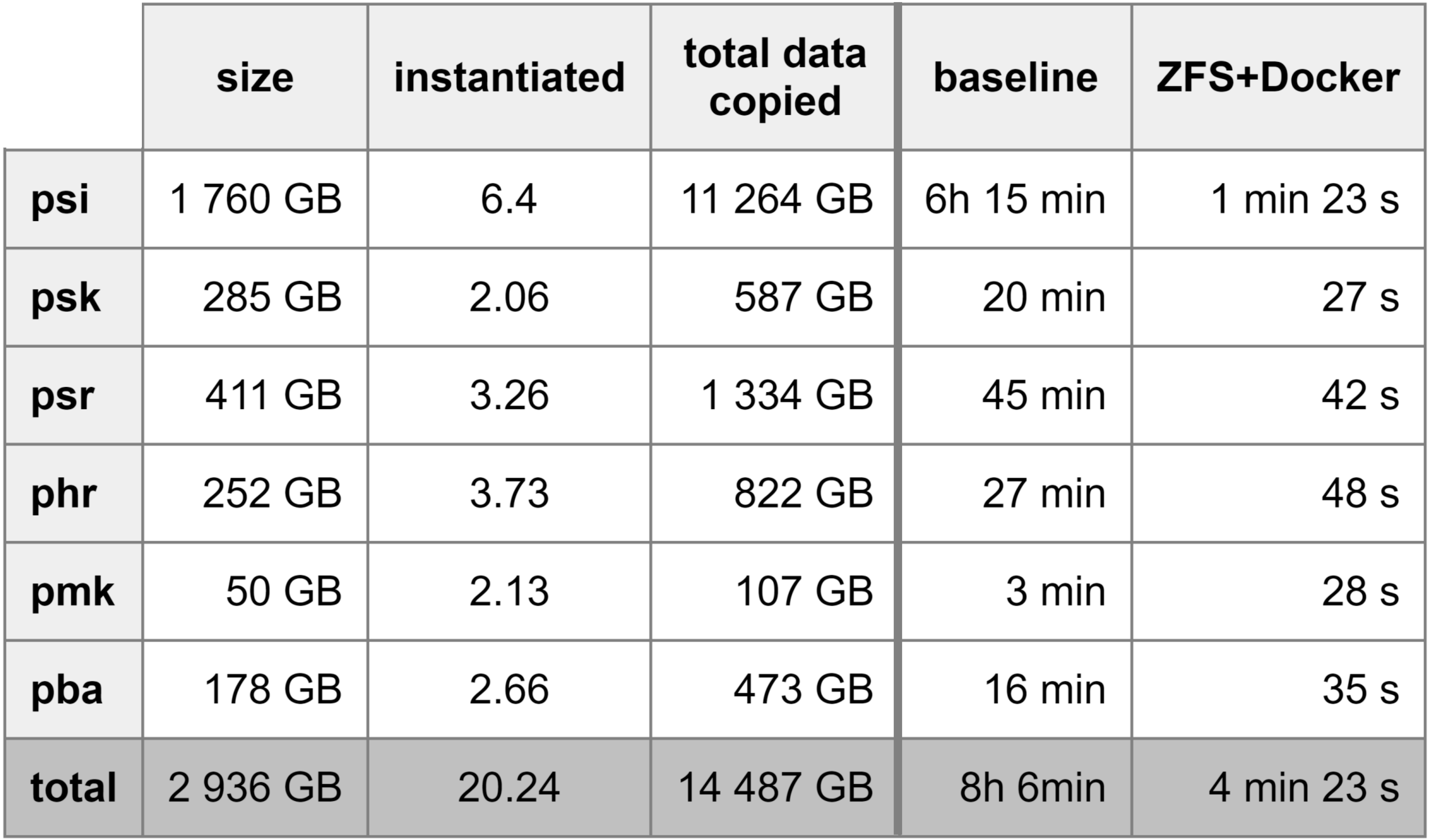
Research objective

- How fast is this system?
- Compare it to a “baseline”: regular database copy&paste

- Is the proposed solution more time-space efficient?


Instance startup timestamp
Wednesday 21 Jun
Tuesday 20 Jun
Monday 19 Jun
Sunday 18 Jun
Saturday 17 Jun
Friday 16 Jun
Thursday 15 Jun
Wednesday 14 Jun
Tuesday 13 Jun
Monday 12 Jun
Sunday 11 Jun
Saturday 10 Jun
Friday 09 Jun
Thursday 08 Jun
Wednesday 07 Jun
Tuesday 06 Jun
Monday 05 Jun
Sunday 04 Jun
Saturday 03 Jun
Friday 02 Jun
Thursday 01 Jun
pba
phr
pmk
psi
psk
psr
Database name

Daily averages for all databases



	size	instantiated	total data copied	baseline	ZFS+Docker
psi	1 760 GB	6.4	11 264 GB	6h 15 min	1 min 23 s
psk	285 GB	2.06	587 GB	20 min	27 s
psr	411 GB	3.26	1 334 GB	45 min	42 s
phr	252 GB	3.73	822 GB	27 min	48 s
pmk	50 GB	2.13	107 GB	3 min	28 s
pba	178 GB	2.66	473 GB	16 min	35 s
total	2 936 GB	20.24	14 487 GB	8h 6min	4 min 23 s

Conclusion

- Orders of magnitude faster
- Saves time
- Saves space
- Scalable

Thank you for your attention