NVMM Cache design: Logging vs. Paging

Work in progress paper: A comparison between two caching mechanisms

Rémi Dulong[1,2], Quentin Acher[3], Baptiste Lepers[1], Valerio Schiavoni[1], Pascal Felber[1], Gaël Thomas[2]

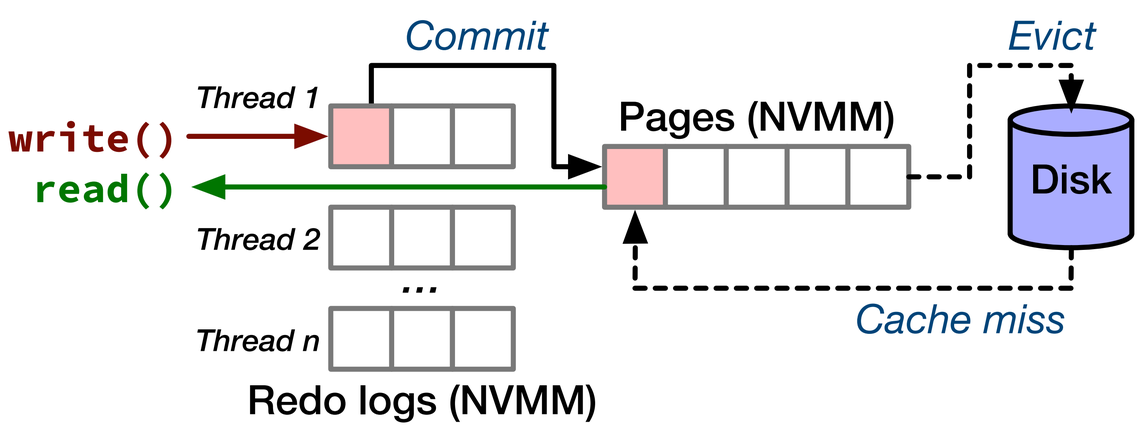
[1]University of Neuchâtel, Switzerland [2]Télécom SudParis, Evry, France [3]ENS Rennes, France

Context

Modern NVMM is closing the gap between DRAM and persistent storage, both in terms of performance and features. Having both byte addressability and persistence on the same device gives NVMM an unprecedented set of features, leading to the following question:
How should we design an NVMM-based caching system to fully exploit its potential?

We built and compared two caching mechanisms, ***NVPages*** and ***NVLog***, based on two radically different design approaches.

NVPages

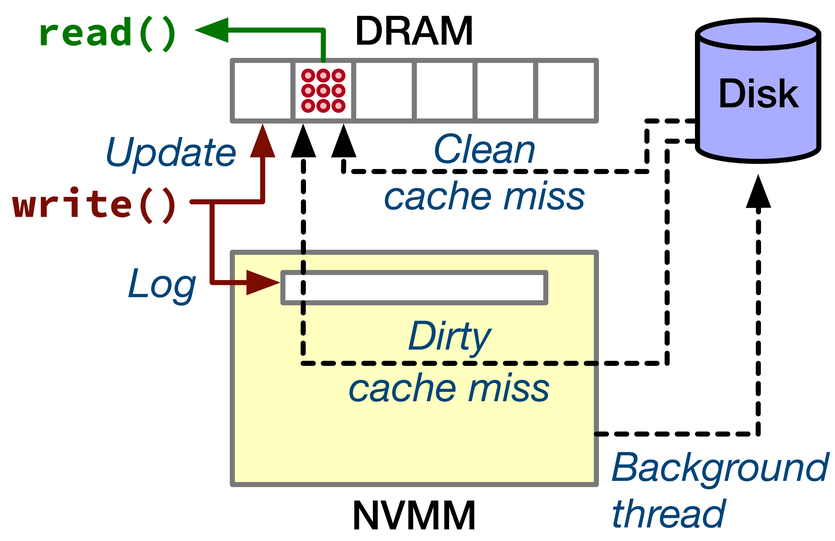


Design:
- One page cache in NVMM, used for reads and writes
- One redo-log per thread
- Inspired from the *Linux Page Cache*

Pros:
- Provides several GiB of non-volatile pages
- No DRAM footprint
- Simple design

Cons:
- Does not benefit from DRAM bandwidth

NVLog



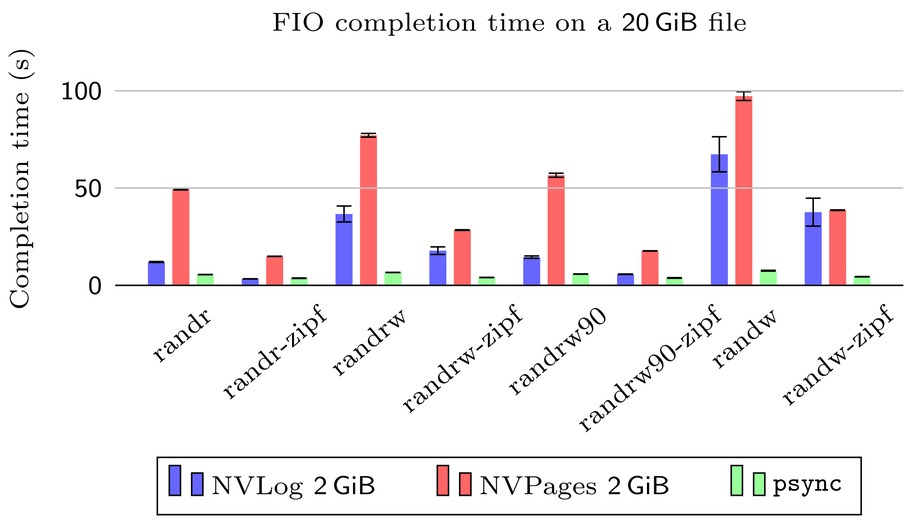
Design:
- NVMM is used as a Log of pending writes
- A background thread writes changes to disk
- A small DRAM page cache keeps hot pages updated

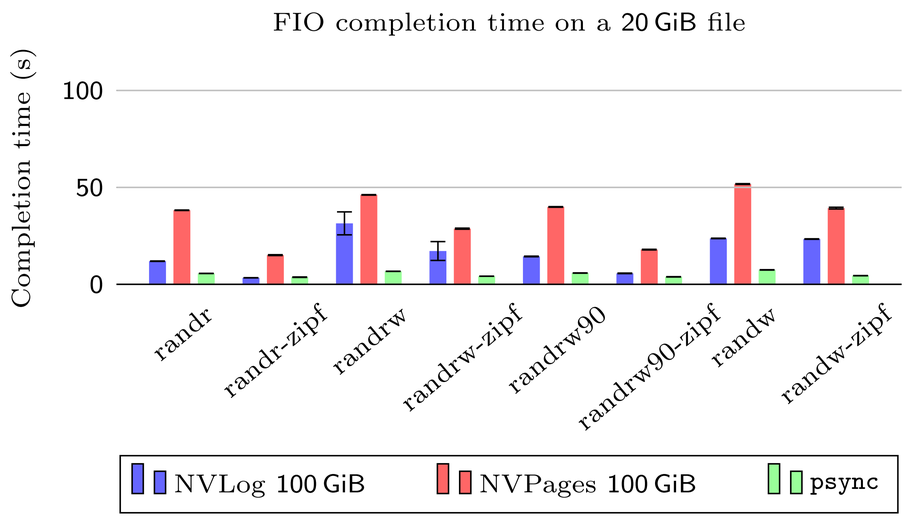
Pros:
- Writes in NVMM, reads in DRAM
- Small DRAM footprint (a few GiB)

Cons:
- Complex design (*Dirty cache miss*, synchronization…)

Inspired from NVCache (DSN'21): https://ieeexplore.ieee.org/document/9505164

Evaluation





Baseline: The "psync" IO engine of FIO uses regular pread and pwrite operations, with no guarantee of persistence.

NVMM Caches

Why should we consider NVMM caches?

- **High persistence guarantees** & Crash resilience
- Compatibility with legacy file systems
- Does not limit storage capacity to NVMM capacity
- **Simpler** than a hybrid file system

What should a NVMM cache provide?

- A transparent POSIX-like interface
- High capacity with NVMM **bandwidth** & **latencies**

Conclusion

In its current state, **NVLog** seems to have a clear edge over **NVPages**. It performed better on all workloads, even on those we expected NVPages to be more efficient. That said, some additional logic should be added to both caches implementations in order to evaluate their performance on multithread workloads.

Université de Neuchâtel — TELECOM SudParis — Institut Polytechnique de Paris

remi.dulong@unine.ch