Measuring Cache-Replacement Policies in a Reproducible Workload

Jordan Blake Department of Computer Science
Northbridge Systems Laboratory
Northbridge, UK

Technical Report: NB-TR-2026-08, August 2026

Table I. Illustrative miss-rate summary. Replace these values with measured output.

Capacity	Requests	FIFO miss rate	LRU miss rate
4	900	0.61	0.54
8	900	0.38	0.31
16	900	0.22	0.19

***Abstract*—This report compares least-recently-used and first-in-first-out cache replacement on a small, deterministic trace. The starter makes the workload, measurement rule, and limitations explicit so that a technical result can be repeated rather than merely described.**

I. Problem and research question

Which policy gives the lower miss rate when a cache receives a short sequence of repeated reads and a burst of one-time keys? The answer depends on the working-set size, so the experiment varies the cache capacity instead of reporting a single convenient configuration.

II. Method

The benchmark replays a fixed trace of integer keys. Each policy starts with an empty cache, and a hit leaves the key in the cache while a miss evicts the oldest eligible key. No prefetching or timing information is used; this keeps the measurement focused on replacement decisions.

```
{
  "trace": "repeated-then-burst",
  "capacities": [4, 8, 16],
  "policies": ["fifo", "lru"],
  "repetitions": 30
}
```

Listing 1. Example machine-readable run configuration

III. Results

IV. Interpretation and limitations

The gap is largest when the repeated keys remain useful after the burst. A real workload may have a different access distribution, and a miss-rate comparison does not measure latency, write-back cost, or concurrency. Record the exact trace, implementation revision, hardware, and random seed alongside the final table.

V. Conclusion

For this trace, LRU retains useful repeated keys more often than FIFO. The conclusion is conditional on the workload and capacities in Table I; the next experiment should test traces collected from the target application.

References

[1] Jordan Blake. *A reproducible cache benchmark for technical reports*. Northbridge Systems Laboratory. Aug. 2026. URL: https://example.com/cache-benchmark (visited on 2026-08-29).

This work is licensed under a Creative Commons Attribution 4.0 International (CC BY 4.0) license.
Please cite as: Jordan Blake. *A reproducible cache benchmark for technical reports*. Northbridge Systems Laboratory. Aug. 2026. URL: https://example.com/cache-benchmark (visited on 2026-08-29).