User Id: mame23

CSE 303: Quiz #3

Due Friday November 4th, 2022 at 11:59 PM

The quiz has THREE questions. Please submit your answer on CourseSite as a pdf whose name is exactly your user Id and the “pdf” extension (e.g., abc123.pdf) before the deadline.

Question 1: Consider the hash table we had in P2, implemented as an array of buckets. In each bucket, there is a lock and a list of key/value pairs. A pair is located in the bucket by using some hash function on its key to choose the appropriate bucket in the array. Now suppose you want to add a `` `move` `` operation to this hash table, defined as follows:

```
/// Given a key k1 and a key k2, if there exists a mapping from k1 to some value v, and there is no
mapping from k2 to any value, then eliminate k1's mapping, and create a new mapping from k2 to v.
///
/// NB: This method must appear to be atomic from the perspective of concurrent insert, remove,
upsert, do_with, do_with_readonly, and do_all_readonly operations
///
/// @return MOVED if the move happens, NOVAL if k1 did not have a mapping in the hash table, and
NOROOM if k2 already had a mapping.
```

Provide a pseudocode for this operation (no need for a full C/C++ code, only a pseudo code that highlights the main functionality). **It must handle important safety and progress pathologies.**

```
Hash key k1 by the size of the hash table to get the index of k1
Create a lock, l, from the index of k1
Loop to k1
    Check if k1 has a value associated with it
        If val(k1) != null
            If val(k2) = null
                Move the pointer or head of k1 to point to the value v2
                Change pointer or head of k2 to null
                Return MOVED
            Else if val(k2) != null
                Return NOROOM
        Else
            Return NOVAL
Return false;
```

User Id: mame23

Question 2: What is one thing you learned about persistence in CSE 303 that you didn't know before?

Please use the remainder of this page to provide your answer to the question. Give a detailed answer but keep your entire response to a single side of an 8.5x11 page.

What I understood before in ECE201 is the principle of locality where the memory hierarchy allows for faster, smaller, and more cost-effective persistent devices. What I learned in CSE303 about persistence is about the power of layering and how the application goes through multiple steps in saving data. At the top layer are the Application, library and File system, where convenient system calls are implemented to enable buffering and prefetching and to avoid the tedious and time-consuming system calls. The middle layer of file systems and device drivers include the file system, block cache (where synchronization through fsync is handled), block device interface, and the device drivers themselves. The final layer to execute the persistence is through the memory mapped I/o and the physical memory device. The technology of Direct Memory Access is also used here.

Question 3: Caching and prefetching are two techniques that are frequently employed in enterprise storage systems. Define and contrast the two techniques, making sure to give examples to illustrate both the strengths and weaknesses of each.

Please use the remainder of this page to provide your answer to the question. Give a detailed answer, but keep your entire response to a single side of an 8.5x11 page.

A cache is a digital intermediate storage that holds already accessed data for quick reuse.[1] A cache is used to hold data that is often required. It would be a waste of time to keep accessing the data from a lower-level storage device. Instead, the data that is often required is held closer to the CPU for quick access times. A cache also holds similar bytes of data. A clear advantage of caches is the increase in speed; this increase is only realized if the data is being accessed repeatedly. Another advantage of caches is the low stress on the underlying system. Accessing a webpage directly from a database is time consuming and eats up bandwidth. Implementing a caching system allows for more throughput. A disadvantage of a cache is invalidation. What method does the operating system use to say, "this data in the cache has not been used in a while/is no longer up-to-date?" This problem is further exacerbated by the potential for a cache miss; We want to avoid them whenever possible.

Prefetching is loading a resource before it is required to potentially decrease waiting time, shown in instruction prefetching in a CPU or a web browser prefetching commonly accessed web page. Prefetching implements caches. A potential downside is the risk of wasting time by prefetching data that may never be used. "Prefetching suffers from certain disadvantages such as an increase in memory traffic, an increase in the number of executed instructions, and an increase in memory requirement for some cases."[2] There is a security vulnerability in prefetching data that is not used, accessing data that is irrelevant to the data needed. A potential advantage to prefetching data is the cost effectiveness of gathering large data requests because large requests are cheaper per byte than smaller ones. Another advantage to prefetching is if the storage device supports parallel I/o, then there is no impact on concurrency.

[1] https://www.ionos.com/digitalguide/hosting/technical-matters/what-is-a-cache/
[2] https://dl.acm.org/doi/10.1145/773039.512450