5.17 Concluding remarks

This section has been set as optional by your instructor.

The difficulty of building a memory system to keep pace with faster processors is underscored by the fact that the raw material for main memory, DRAMs, is essentially the same in the fastest computers as it is in the slowest and cheapest.

It is the principle of locality that gives us a chance to overcome the long latency of memory access—and the soundness of this strategy is demonstrated at all levels of the **memory hierarchy**. Although these levels of the hierarchy look quite different in quantitative terms, they follow similar strategies in their operation and exploit the same properties of locality.

Multilevel caches make it possible to use more cache optimizations more easily for two reasons. First, the design parameters of a lower-level cache are different from a first-level cache. For example, because a lower-level cache will be much larger, it is possible to use bigger block sizes. Second, a lower-level cache is not constantly being used by the processor, as a first-level cache is. This allows us to consider having the lower-level cache do something when it is idle that may be useful in preventing future misses.

Another trend is to seek software help. Efficiently managing the memory hierarchy using a variety of program transformations and hardware facilities is a major focus of compiler enhancements. Two different ideas are being explored. One idea is to reorganize the program to enhance its spatial and temporal locality. This approach focuses on loop-oriented programs that use sizable arrays as the major data structure; large linear algebra problems are a typical example, such as DGEMM. By restructuring the loops that access the arrays, substantially improved locality—and, therefore, cache performance—can be obtained.

Another approach is *prefetching*. In prefetching, a block of data is brought into the cache before it is actually referenced. Many microprocessors use hardware prefetching to try to **predict** accesses that may be difficult for software to notice.

Prefetching: A technique in which data blocks needed in the future are brought into the cache early by using special instructions that specify the address of the block.

A third approach is special cache-aware instructions that optimize memory transfer. For example, the microprocessors in COD Section 6.10 (Multiprocessor benchmarks and performance models) use an optimization that does not fetch the contents of a block from memory on a write miss because the program is going to write the full block. This optimization significantly reduces memory traffic for one kernel.

As we will see in COD Chapter 6 (Parallel Processor from Client to Cloud), memory systems are a central design issue for parallel processors. The growing significance of the memory hierarchy in determining system performance means that this important area will continue to be a focus for both designers and researchers for some years to come.

PARTICIPATION ACTIVITY 5.17.1: Concluding remarks.

1) A cache exploits the principles of ____ to overcome the latency associated with memory access.

Check Show answer

2) ____ is a method of predicting and fetching data blocks that are needed in the future.

Check Show answer

Provide feedback on this section