Veljko Milutinovic
University of Belgrade

Caching in Distributed Systems

Modern computer systems, as well as the Internet, use caching to maximize their efficiency. Nowadays, caching occurs in many different system layers. Analysis of these layers will lead to a deeper understand of cache performance.

Caching

Cache memory is a small but fast memory meant to hold data for reuse in the near future. Maurice V. Wilkes introduced the concept to fulfill the speed gap between the CPU and main memory.[1] He envisioned that cache memory would bridge that gap by using the principles of spatial and temporal locality.

Recent advances in computer systems engineering have pushed cache memory to higher levels in the computer systems hierarchy. On each new level, the implementation details differ (to reflect the concrete requirements of the particular system level), but the essence stays the same (to reflect the chosen methods for using the principles of spatial and temporal locality). So, the principles of spatial and temporal locality help the concept survive and spread into the newly opened layers of the emerging computer system hierarchies.

The seven layers of caching

A careful analysis on various system levels in current systems reveals seven layers of caching:

1. CPU (in uniprocessor systems),
2. SMP (in shared memory multiprocessor systems),
3. DSM (in distributed shared memory systems),
4. DFM (in distributed file management and smart disk systems),
5. DPC (in distributed proxy cache systems),
6. WWW (on the World Wide Web level), and
7. IAI (on the Internet application and integration level).

In principle, this number could be higher (if a higher granularity of system analysis is implied) or lower (because different caching layers are highly correlated). We can define the principles of spatial and temporal locality on each layer.

CPU

The traditional definition of spatial and temporal localities comes from the uniprocessor environment. Spatial locality implies that the *next* data item in the address space is most likely to be used next, while temporal locality implies that the *last* data item used is most likely to be used next. Implementation is typically based on a fast but expensive memory (the price is affordable because, by definition, cache memory is small). Even if we use the same technology for the main memory and cache memory, the cache memory will be faster because smaller memories have a shorter access time. Recent research tries to split the CPU cache into two subcaches: one for spatial locality and one for temporal locality.[2]

SMP

On the SMP level, spatial and temporal

1092-3063/00/$10.00 © 2000 IEEE

In this issue

"Splitting the Data Cache," by Julio Sahuquillo and Ana Pont (p. 28) , sheds light on split spatial and temporal caches. It is oriented to the single-processor domain, but it covers important issues for all seven layers of caching.

"Achieving High Performance in Bus-Based Shared-Memory Multiprocessors," by Aleksandar Milenkovic (p. 34), discusses the concept of cache injection. Again, the article presents the concept in a shared-memory multiprocessor domain, but it applies to all higher layers of caching.

"Delta Coherence Protocols," by Craig Williams (p.21) , Paul F. Reynolds, Jr., and Bronis de Supinski, describes a class of directory coherence protocol. These delta coherence protocols are characterized by a number of features for modern distributed shared memory systems.

In "Hierarchical Caching and Prefetching for Continuous Media Servers" (p. 14), Stavros Harizopoulos, Costas Harizakis, and Peter Triantafillou propose new caching and prefetching algorithms for media servers with smart disks. Their goal is to increase the maximum number of streams that a drive can support.

"The Differences between Distributed Shared Memory Caching and Proxy Caching," by Juan-Carlos Cano, Ana Pont, Julio Sahuquillo, and José-Antonio Gil (p. 39), sheds more light on proxy caching. It revisits several relevant issues for cache design, comparatively for proxy caches and caches in multiprocessors (specifically DSM). The goal is to emphasize essential differences to avoid misunderstandings and wrong design decisions.

In "A Scalable and Efficient Cooperative System for Web Caches" (p. 42), Jean-Marc Menaud, Valérie Issarny, and Michel Banâtre propose a novel algorithm that enables more accurate replacement decisions, as well as a cooperation protocol that minimizes the most relevant design requirements: network bandwidth, processing load, and storage consumption among caches.

"Cache Management in CORBA Distributed Object Systems," by Zahir Tari, Qi Tang Lin, and Herry Hamidjaja (p. 49), introduces a caching approach for the CORBA environment.

locality continue to be present on the uniprocessor level. However, on the multiprocessor level, new forms of locality gain importance: processor locality, locality of shared data, and so forth. Implementation also includes mechanisms for maintaining data consistency, on either the hardware or software levels. Recent research concentrates on cache miss and bus traffic reduction by combining conventional and new approaches, such as the prefetch and injection approaches.[3]

DSM

Caches on the DSM level also exist on the CPU and SMP levels (a DSM system often consists of clusters implemented as SMP systems). Misses on the DSM level can be extremely costly; however, the caches on the DSM level have much more difficulty capturing locality.

DFM

On the DFM level, caches can help implement several different applications (media servers, file distribution, and so forth). Spatial locality is present much more than temporal locality. Additional types of locality, stemming from the specific internal and external disk structure, can also be defined and used for better system efficiency. Recent research concentrates on the so-called smart disks, using different specialized resources to maximize performance.

DPC

The DPC level uses caching in conjunction with protection. In addition to spatial locality (present to a smaller extent) and temporal locality (present to a larger extent), we can define and use many different types of locality: URL, geographical (if it can be defined), user, institutional, and so forth. Often, distinguishing the specifics of the distributed proxy cache is difficult; consequently, strategic errors in proxy cache design are possible.

WWW

On the WWW level, caches subdivide into client, server, and network caches. The client cache's primary goal is to handle data reusability, which improves the Web latency. The server cache's primary goal is to reduce the server node workload, and the network cache's primary goal is to help clients benefit from the earlier accesses (to the same data) by other clients sharing the network cache. Types of locality in the three cache subtypes are the subject of ongoing research. Current implementations concentrate on problems in the domain of cache management (for example, replacement protocols) and cache cooperation (cooperation protocols).

IAI

On the IAI level, we try to detect reusability and use the principles of locality in the systems responsible for Web-oriented application and integration software. The existing types of locality depend on the specific types of software in use. Concrete implementations differ greatly and mainly concentrate on the issues of importance for cache management and cache replacement.

THE SEVEN ARTICLES I SELECTED for this special issue (see the "In this issue" sidebar) represent important problems related to the seven layers of caching I advocate in this overview.

References

1. M.V. Wilkes, "Slave Memories and Dynamic Storage Allocation," *Trans. IEEE*, Vol. EC-14, No. 2, Apr. 1965, pp. 270–271.
2. V, Milutinovic et al., "The Split Temporal/Spatial Cache: Initial Performance Analysis," *Proc. SCIzzl-5*, Mar. 1996, pp. 72–78.
3. V. Milutinovic, A. Milenkovic, and G. Shaeffer, "The Cache Injection Control Architecture: Initial Performance Analysis," *Proc. Mascots '97: Fifth Int'l Symp. Modeling, Analysis, and Ssimulation of Computer and Telecommunications Systems*, IEEE Computer Soc. Press, Los Alamitos, Calif., 1997, pp. 63–64.

Veljko Milutinovic is a professor of computer engineering at the University of Belgrade. His research interests include computer architecture and design and the infrastructure for e-business on the Internet. He received his BS, MSc, and PhD from the University of Belgrade. He has been a consultant for numerous high-tech companies in the US, the Far East, and Europe. Contact him at vm@etf.bg.ac.yu; http://galeb.etf.bg.ac.yu/~vm.