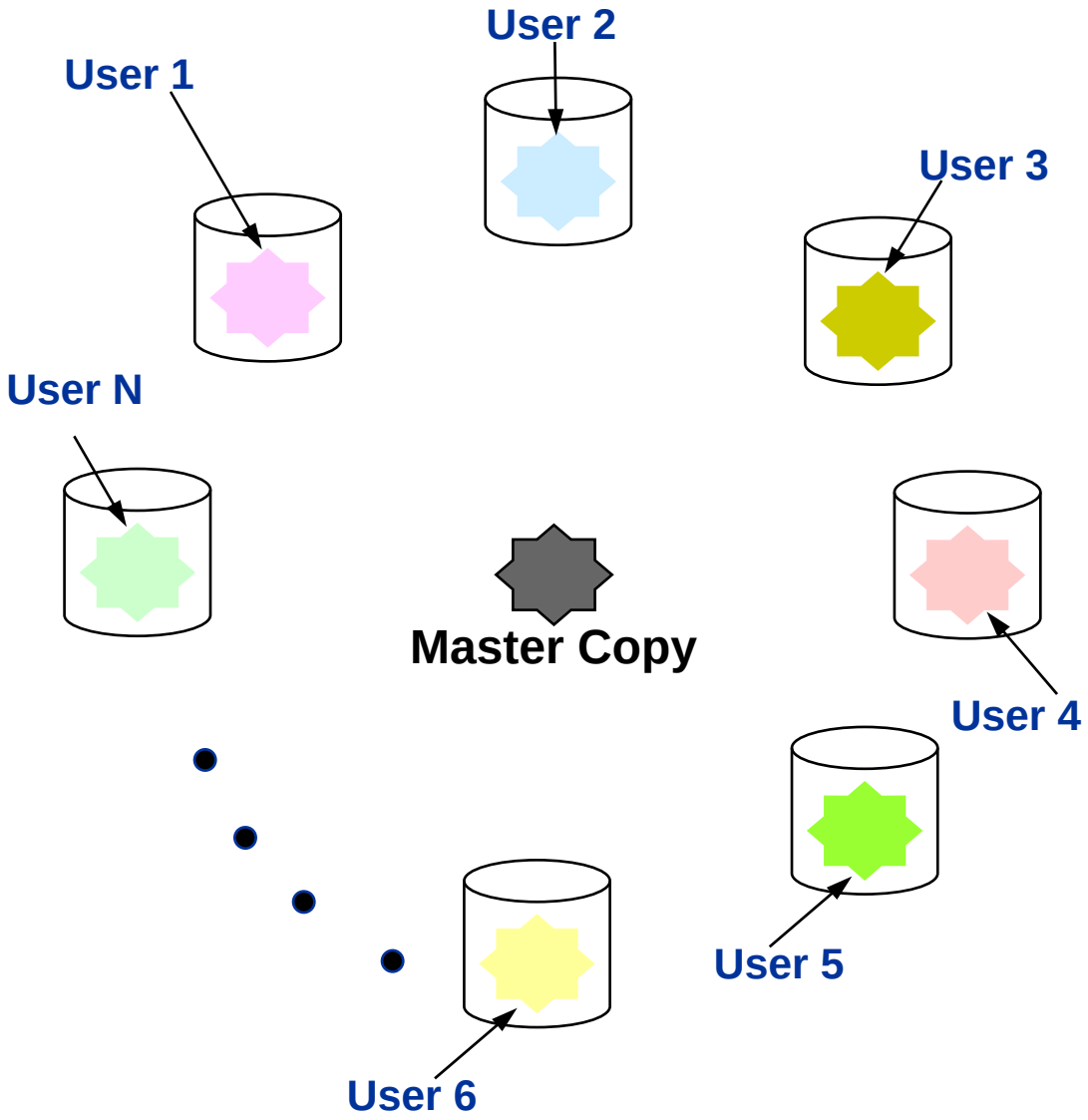
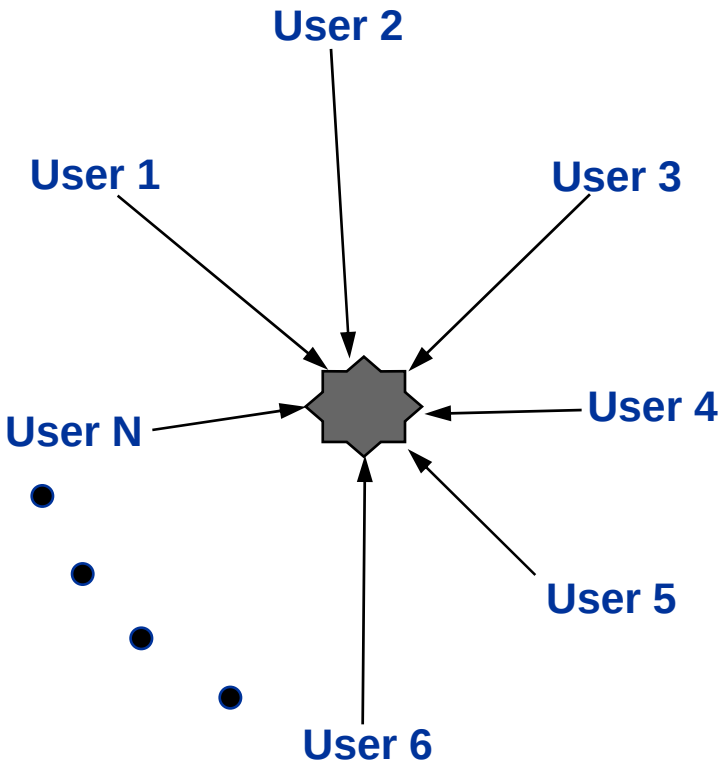
Update Propagation in a Cache-based System

(aka "Cache Consistency")

Caching Reality



Desired Illusion



One-copy semantics

What Makes This Hard?

Physical master copy may not exist

- hosts track who has most recent copy
- more likely in P2P scenarios than client-server scenarios
- also common in multiprocessor hardware caches

Network may break between some users and master copy

- disconnected sites see no further updates
- other sites don't see updates by disconnected site

Intense read- and write-sharing across sites

- generates huge amount of cache propagation traffic
- interconnect becomes bottleneck for cache access
- neither writer-bias (write-back) nor reader-bias (write-through) helps
- caches effectively useless

Required Readings

- **[Howard88]**
  Howard, J. H., Kazar, M. L., Menees, S. G., Nichols, D. A., Satyanarayanan, M., Sidebotham, R. N., and West, M. J. 1988. Scale and performance in a distributed file system. *ACM Trans. Comput. Syst.* 6, 1 (Feb. 1988)

- **[Gray89]**
  Gray, C. and Cheriton, D. 1989. Leases: an efficient fault-tolerant mechanism for distributed file cache consistency. In *Proceedings of the Twelfth ACM Symposium on Operating Systems Principles*, Litchfield Park, AZ

Optional Readings

- **[Strenstrom90]**
  P. Stentrsom, "A Survey of Cache Coherence Schemes for Multiprocessors", IEEE Computer, June 1990
- **[Satya85]**
  Satyanarayanan, M., Howard, J. H., Nichols, D. A., Sidebotham, R. N., Spector, A. Z., and West, M. J. 1985. The ITC distributed file system: principles and design. In *Proceedings of the Tenth ACM Symposium on Operating Systems Principles,* Orcas Island, WA
- **[Nelson88]**
  Nelson, M. N., Welch, B. B., and Ousterhout, J. K. 1988. Caching in the Sprite network file system. *ACM Trans. Comput. Syst.* 6, 1 (Feb. 1988)
- **[Luotonen94]**
  Luotonen, A. and Altis, K. 1994. World-Wide Web proxies. In *Selected Papers of the First Conference on World-Wide Web* (Geneva, Switzerland). R. Cailliau and P. H. Enslow, Eds. Elsevier Science Publishers B. V., Amsterdam, The Netherlands
- **[Franklin97]**
  Franklin, M. J., Carey, M. J., and Livny, M. 1997. Transactional client-server cache consistency: alternatives and performance. *ACM Trans. Database Syst.* 22, 3 (Sep. 1997)
- **[Yin02]**
  Yin, J., Alvisi, L., Dahlin, M., and Iyengar, A. 2002. Engineering web cache consistency. *ACM Trans. Inter. Tech.* 2, 3 (Aug. 2002),

1. Broadcast Invalidations

Basic Idea

Every potential caching site notified on every update

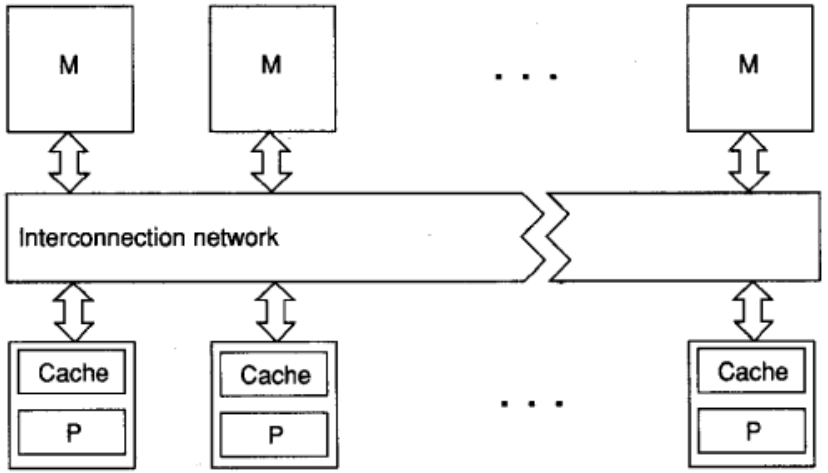
- **No check to verify caching site actually contains object**
- **Notification includes specific object being invalidated**
- **Effectively broadcast of address being modified**

At each cache site, next reference to object will cause a miss

Original Use

- **Small-scale multiprocessors in early 1970's**
- **"snoopy cache" multiprocessor variants of 1980s-1990s**

Reference: *Stenstrom90*



Strengths

- very strict emulation of “one-copy semantics”
- no race conditions if updater blocked until all caches invalidated
- simple to implement

Limitations

- wasted traffic if no readers elsewhere
- updating process blocked until invalidation complete
- not a scalable design

2. Check on Use

Basic Idea

- ***reader checks master copy before each use***
- **conditional fetch, if cache copy stale**
- **typically done at coarse granularity (e.g. entire file)**

Original Use

- **AFS-1 (circa 1983)**
- **Reference: *Satya85***

Advantages

- **Strict consistency at coarse granularity**
- **Easy to implement, no server state**
- **Servers don't need to know of caching sites**

Disadvantages

- **Slows read access on loaded servers & high-latency networks**
- **Check is almost always success → frivolous traffic**
- **Load on network and server**

3. Callback

Basic Idea

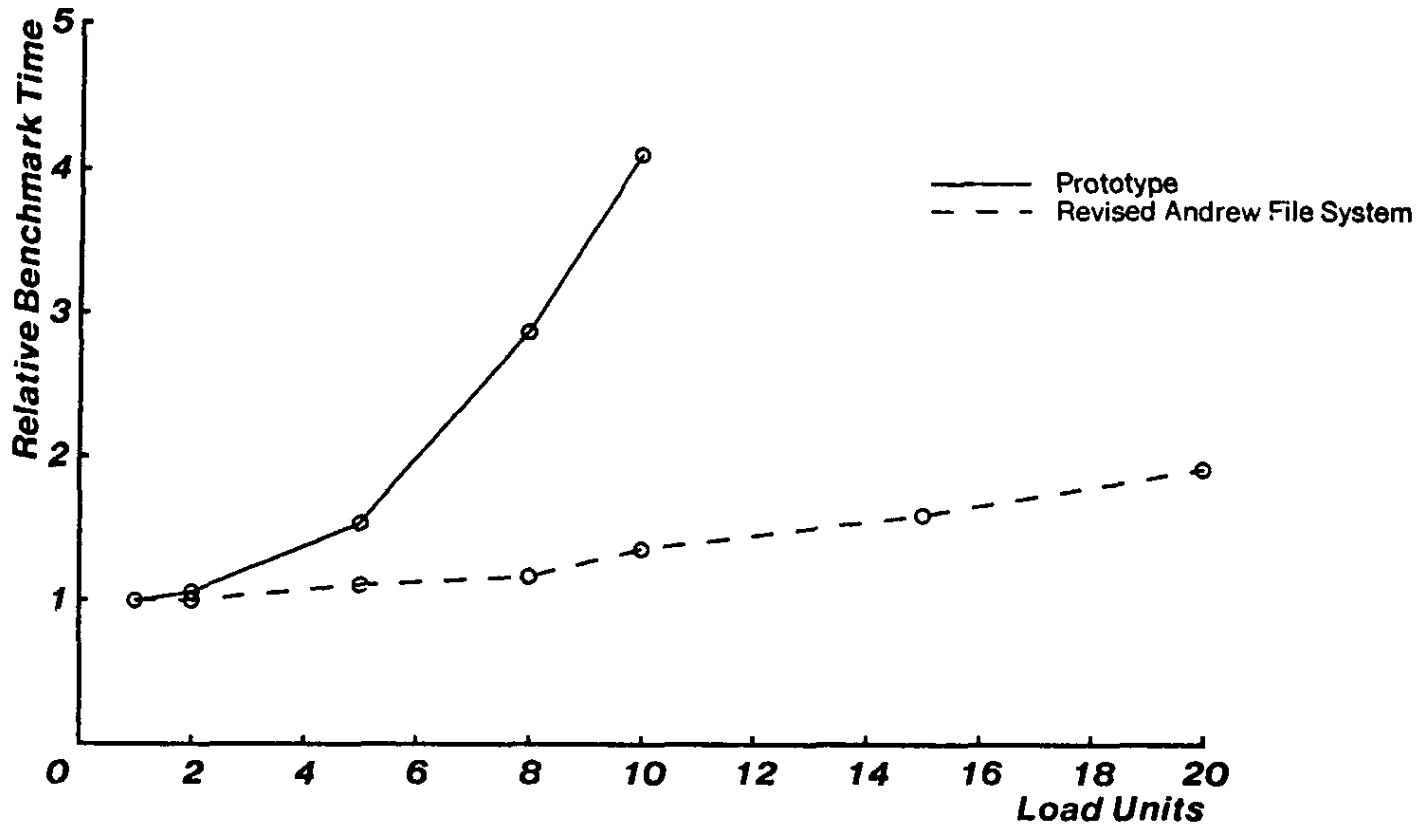
- *targeted notification of caching sites*
- master copy tracks sites with cached data
- typically done at coarse granularity (e.g. entire file)
- on update, all sites with cached data notified *("callback")*

Original Use

- AFS-2 (circa 1985)
- Essentially the same idea in MP systems → *"Directory scheme"*
- Reference: *Howard88, Stenstrom90*

Advantages

- Excellent scalability for Unix workloads
- *Zero network traffic for read of cached-valid objects*
- Precursor to caching for disconnected operation
- Biases read performance in favor of write-performance



Disadvantages

- sizable state on server
- complexity of tracking cached state on clients
- silence ambiguous for client: network failure → lost callbacks
  - periodic "keepalive" probes
  - data could be stale between probes
- NAT networks with masquerading firewalls