Leases: An Efficient Fault Tolerant Mechanism for Distributed File Cache Consistency

Gray and Cheriton

By Farid Merchant
Date: 9/21/05

Problems with Distributed System Caching

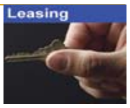
Caching is a mechanism to improve the speed and performance of the system by replicating copies of frequently accessed data. But with the added cost of :

- Complexity of ensuring consistency
  - With large number of clients. One client can modify data without consent of other client, leading other clients to provide stale data to its user.
- Complication of communication and host failures
  - Message lost due to failures

Goals of the presentation

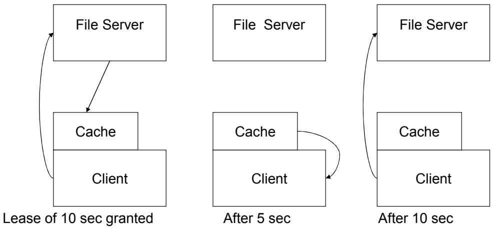
- Introduction to leases
- How leases are implemented for cache consistency
- An Analytical model for lease term evaluation
- Optimization in lease management
- Related work on distributed cache consistency
- Conclusion

Leasing

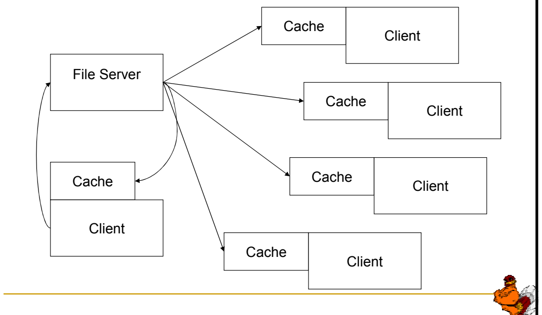


- In General Term
  A lease is a contract that gives its holder specified rights over property for a limited period of time
- In Caching Term
  A lease is a promise by the server that it will push updates to the client as well as grants control over writes to the covered datum for a specified time

Example: Operation of file cache using leases for read



Example: Operation of file cache using leases for write



Implementation of Leases

- A lease is granted to the data when fetched from the server for the first time
- Subsequent fetch from cache during lease
- Server needs approval of the leaseholder
- After lease expiration, it needs to be renewed to access data as well as data should be updated if modified

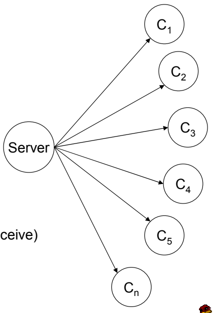
Simple Analytical model

Effective term at Cache

- $t_c = \max(0, t_s - (m_{prop} + 2m_{proc}) - \varepsilon)$

Time to gain approval

- $t_a = 2m_{prop} + (S + 2)m_{proc}$



Symbols → Descriptions
S → number of caches in which the file is shared
m(prop) → propagation delay for the message
m(proc) → time to process a message (send or receive)
ε → allowance for uncertainty in clocks
t_s → lease term (at server)

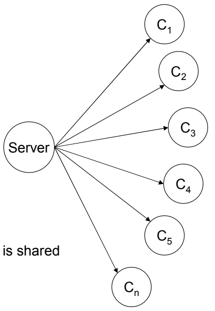
Simple Analytical Model (Cont..)

- Load at the server

$$\frac{2NR}{1+Rt_C} + NSW$$

- Average delay due to each read or write

$$\frac{1}{R+W}\left(\frac{2R(m_{prop}+2m_{proc})}{1+Rt_C} + Wt_a\right)$$



S → number of caches in which the file is shared
N → number of clients (caches)
R → Rate of reads for each client
W → Rate of writes for each client

Simple Analytical Model (Cont..)

- If $t_C > \frac{1}{R(\alpha - 1)}$,
  where $\alpha = \frac{2R}{SW}$
  then

$$2NR > \frac{2NR}{1+Rt_C} + NSW$$

2NR → Load due to zero lease term

- Larger values of α and R implies better performance for short term lease

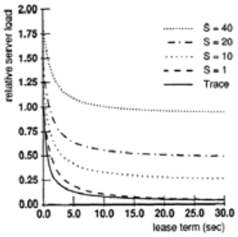


Figure 1: Relative Server Consistency vs. Lease Term

Simple Analytical Model (Cont..)

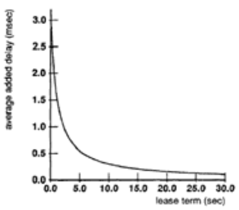


Figure 2: Delay due to consistency.

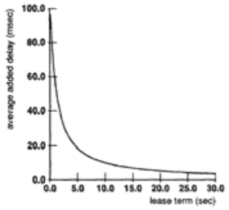
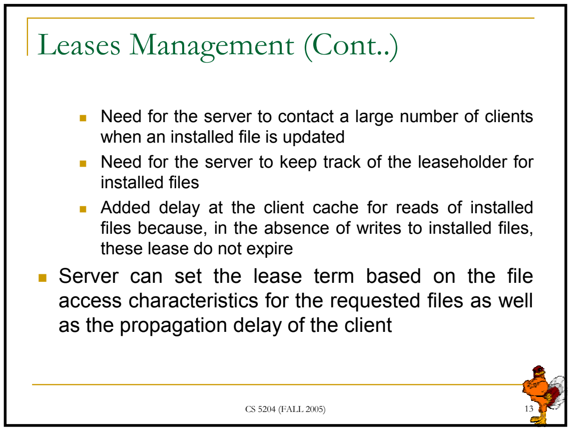


Figure 3: Added delay with 100 ms round-trip time.

Lease Management

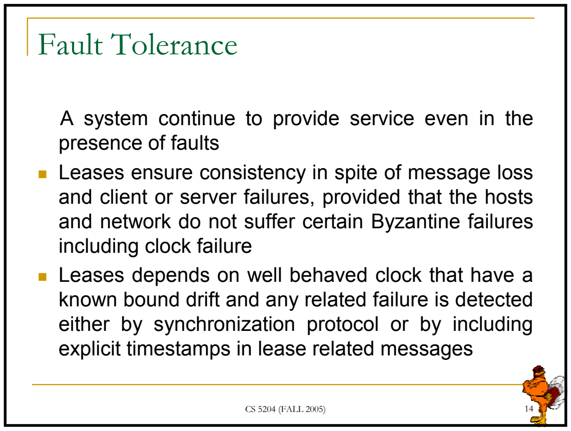
- Request lease extension before the covered file is accessed
  - Improves responses time by eliminating the added delay for reads but with the cost of increased level of contention due to false sharing
- By using smaller number of leases to cover installed files such as one per major directory, and multicasting an extension to all clients periodically eliminates …
  - Need for clients to request extensions of these lease.
  - A lease from the multicast extension when a file covered by the lease is to be modified by the server



Leases Management (Cont..)

- Need for the server to contact a large number of clients when an installed file is updated
- Need for the server to keep track of the leaseholder for installed files
- Added delay at the client cache for reads of installed files because, in the absence of writes to installed files, these lease do not expire

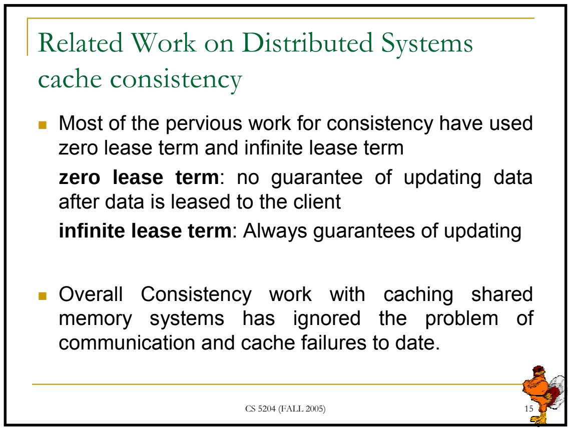
- Server can set the lease term based on the file access characteristics for the requested files as well as the propagation delay of the client



Fault Tolerance

A system continue to provide service even in the presence of faults

- Leases ensure consistency in spite of message loss and client or server failures, provided that the hosts and network do not suffer certain Byzantine failures including clock failure
- Leases depends on well behaved clock that have a known bound drift and any related failure is detected either by synchronization protocol or by including explicit timestamps in lease related messages



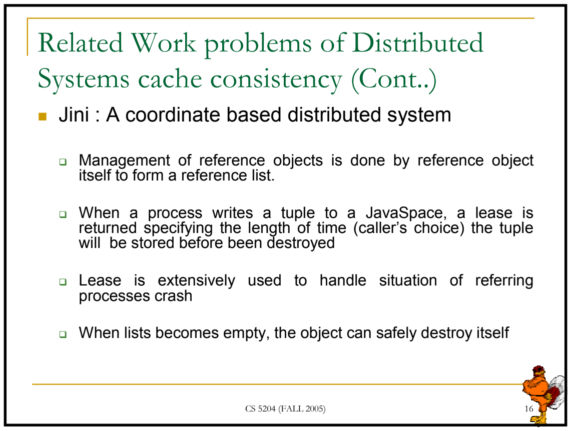
Related Work on Distributed Systems cache consistency

- Most of the pervious work for consistency have used zero lease term and infinite lease term

  zero lease term: no guarantee of updating data after data is leased to the client

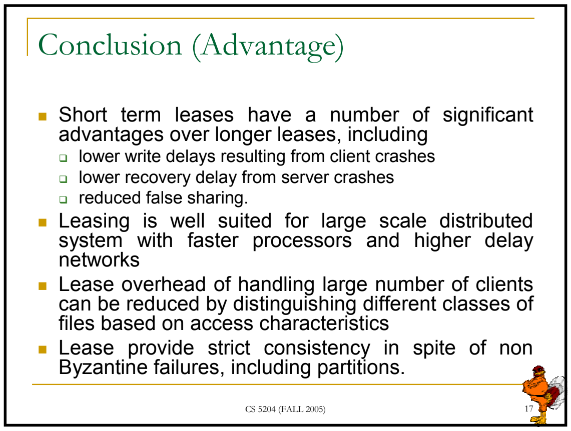
  infinite lease term: Always guarantees of updating

- Overall Consistency work with caching shared memory systems has ignored the problem of communication and cache failures to date.



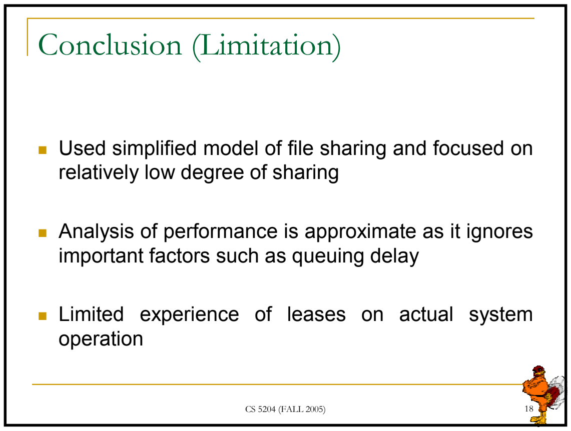
Related Work problems of Distributed Systems cache consistency (Cont..)

- Jini : A coordinate based distributed system
  - Management of reference objects is done by reference object itself to form a reference list.
  - When a process writes a tuple to a JavaSpace, a lease is returned specifying the length of time (caller's choice) the tuple will be stored before been destroyed
  - Lease is extensively used to handle situation of referring processes crash
  - When lists becomes empty, the object can safely destroy itself



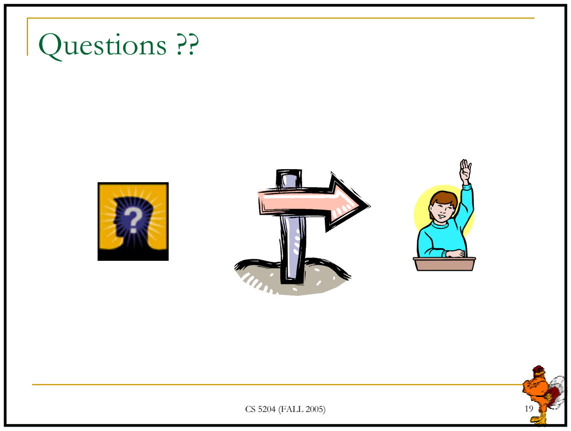
Conclusion (Advantage)

- Short term leases have a number of significant advantages over longer leases, including
  - lower write delays resulting from client crashes
  - lower recovery delay from server crashes
  - reduced false sharing.
- Leasing is well suited for large scale distributed system with faster processors and higher delay networks
- Lease overhead of handling large number of clients can be reduced by distinguishing different classes of files based on access characteristics
- Lease provide strict consistency in spite of non Byzantine failures, including partitions.



Conclusion (Limitation)

- Used simplified model of file sharing and focused on relatively low degree of sharing
- Analysis of performance is approximate as it ignores important factors such as queuing delay
- Limited experience of leases on actual system operation



Questions ??