HyCache: A Hybrid User-Level File System with SSD Caching

Dongfang Zhao
Department of Computer Science
Illinois Institute of Technology
dzhao8@iit.edu

Ioan Raicu
Department of Computer Science, Illinois Institute of Technology
Math and Computer Science Division, Argonne National Laboratory
iraicu@cs.iit.edu

Abstract

For many decades, a huge performance gap has existed between volatile memory and mechanical hard disk drives. This will be a critical issue with extreme scale computing systems. Although non-volatile memory has been around since the 1990's, mechanical hard disk drives are still dominant due to large capacity and relatively low cost. We have designed and implemented HyCache, a user-level file system that leverages both mechanical hard disk drive cost-effectiveness and solid-state drive performance. We adopted FUSE to deliver a user-level POSIX-compliant file system that does not require any OS or application modifications. HyCache allows multiple devices to be used together to achieve better performance while keeping costs low. An extensive evaluation is performed using synthetic benchmarks as well as real world applications, which shows that HyCache can achieve up to 7X higher throughput and 76X higher IOPS over traditional Ext4 file systems on mechanical hard disk drives.

The Problem: disparity in SSD and HDD

Spinning hard disk (HDD) vs. solid state drive (SSD)
- HDD: cheap ✓, large capacity ✓, slow ✗
- SSD: fast ✓, expensive ✗, small capacity ✗

KEY SPECIFICATIONS OF SOME HARD DRIVES ON THE MARKET

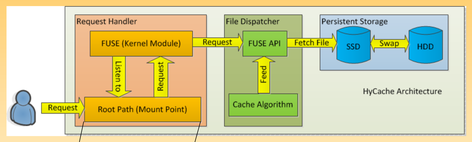
Hard Drive	Unit Price (per GB)	Capacity (GB)	Read (MB/s)	Write (MB/s)	IOPS
OCZ RevoDrive	$3.28	960	1,500	1,300	230,000
OCZ Octane	$1.56	512	480	330	26,000
Seagate Momentus XT	$0.22	504	131	101	238
Hitachi Deskstar	$0.075	4,096	144	142	360

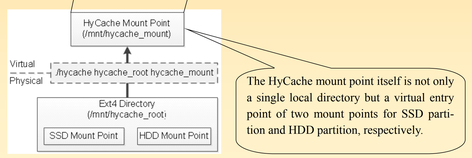
Our Solution: HyCache File System

- **User-level file system**: no change to OS kernel, no change to applications, no privilege needed
- **Fully POSIX-compliant**: no need to learn any new syntax
- **High throughput**: up to 7X bandwidth than Ext4 on HDD
- **Low latency**: up to 76X IOPS than Ext4 on HDD
- **Easy to customize**: plug in your own caching algorithms (LRU and LFU already built-in)

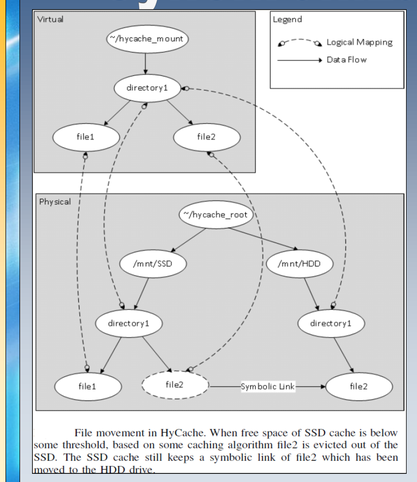
Architecture

- **Request Handler** interacts with end users and passes user requests to file dispatcher.
- **File Dispatcher** takes file requests from request handler and conducts the operations.
- **Persistent Storage** is a mix of high- and slow-speed storage block devices.

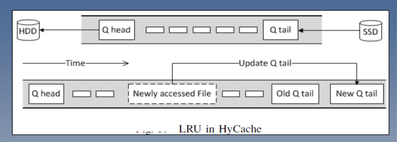




Synchronization of files in SSD and HDD



File movement in HyCache. When free space of SSD cache is below some threshold, based on some caching algorithm file2 is evicted out of the SSD. The SSD cache still keeps a symbolic link of file2 which has been moved to the HDD drive.



LRU in HyCache

Algorithm 1 Open a file in HyCache

Require: F is the file requested by the end user; Q is the cache queue used for the replacement policy; SSD is the mount point of SSD drive; HDD is the mount point of HDD drive
Ensure: F is appropriately opened
1: **if** F is a symbolic link in SSD **then**
2: **while** SSD space is intensive and Q is not empty **do**
3: move some file(s) from SSD to HDD
4: remove these files from the Q
5: **end while**
6: remove symbolic link of F in SSD
7: move F from HDD to SSD
8: insert F to Q
9: **else**
10: adjust the position of F in Q
11: **end if**
12: open F in SSD

Algorithm 2 Remove a file in HyCache

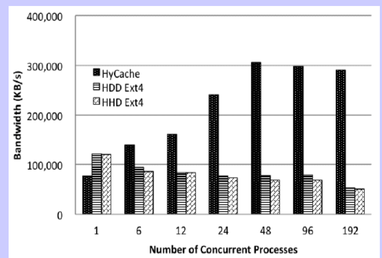
Require: F is the file requested by the end user for removal; Q is the cache queue used for the replacement policy; SSD is the mount point of SSD drive; HDD is the mount point of HDD drive
Ensure: F is appropriately removed
1: **if** F is a symbolic link in SSD **then**
2: remove F from HDD
3: **end if**
4: remove F from Q
5: remove F from SSD

Algorithm 3 Rename a file in HyCache

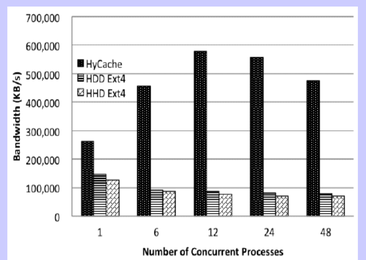
Require: F is the file requested by the end user to rename; F' is the new file name; Q is the queue used for the replacement policy; SSD is the mount point of SSD drive; HDD is the mount point of HDD drive
Ensure: F is renamed to F'
1: **if** F is a symbolic link in SSD **then**
2: rename F to F' in HDD
3: remove F in SSD
4: create the symbolic link F' in SSD
5: **else**
6: rename F to F' in SSD
7: rename F to F' in Q
8: **end if**
9: update F' position in Q

Performance

Bandwidth with IOZone



(a) 4KB Block



(b) 64KB Block

IOPS

IOPS OF DIFFERENT FILE SYSTEMS

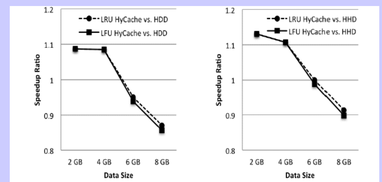
HyCache	HDD Ext4	HHD Ext4
14,878	195	61

Metadata

NUMBER OF METADATA OPERATIONS PER SECOND

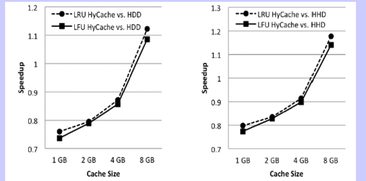
HyCache	HyCache w/ NoVoHT	HDD	HHD
68	322	362	367

PostMark Speedup



(a) HyCache vs. HDD (b) HyCache vs. HHD

PostMark: speedup of HyCache over Ext4 with 4 GB SSD cache.



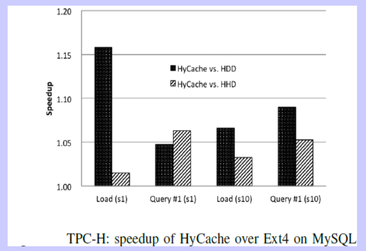
(a) HyCache vs. HDD (b) HyCache vs. HHD

PostMark: speedup of HyCache with varying sizes of cache.

TPC-H / MySQL on HyCache

```
select
    l_returnflag,
    l_linestatus,
    sum(l_quantity) as sum_qty,
    sum(l_extendedprice) as sum_base_price,
    sum(l_extendedprice * (1 - l_discount)) as sum_disc_price,
    sum(l_extendedprice * (1 - l_discount) * (1 + l_tax)) as sum_charge,
    avg(l_quantity) as avg_qty,
    avg(l_extendedprice) as avg_price,
    avg(l_discount) as avg_disc,
    count(*) as count_order
from
    lineitem
where
    l_shipdate <= date '1998-12-01' - interval '72' day
group by
    l_returnflag,
    l_linestatus
order by
    l_returnflag,
    l_linestatus;
```

Fig. 13. TPC-H: Query #1.



TPC-H: speedup of HyCache over Ext4 on MySQL

Conclusion and Future Work

- HyCache is a cost-effective solution to alleviate the storage bottleneck
- This work can be a solid building block for future distributed storage systems, aimed at delivering comparable performance of an all SSD solution at a fraction of the cost
- Our extensive performance evaluation showed that user-level file systems can be competitive with kernel-level file systems as well as embedded hybrid hard drive technologies.
- HyCache will adopt NoVoHT to improve the performance of metadata operations as the default metadata management.
- HyCache will be eventually integrated into FusionFS which is a high-performance distributed file system aimed at exascale computing

References

- SSDAlloc: a SSD extension to main memory, NSDI 2011
- I-CASH: SSD-\mapping random I/O to sequential I/O, HPCA 2011
- Hystor: a kernel-level hybrid SSD+HDD storage, ICS 2011
- HeteroDrive: SSD redundancy buffer for disk arrays, IPDPS 2010
- HyCache website: http://datasys.cs.iit.edu/projects/HyCache/index.html