Reducing Hybrid Disk Write Latency with Flash-Backed I/O Requests

Timothy Bisson and Scott A. Brandt
University of California, Santa Cruz
Santa Cruz, CA

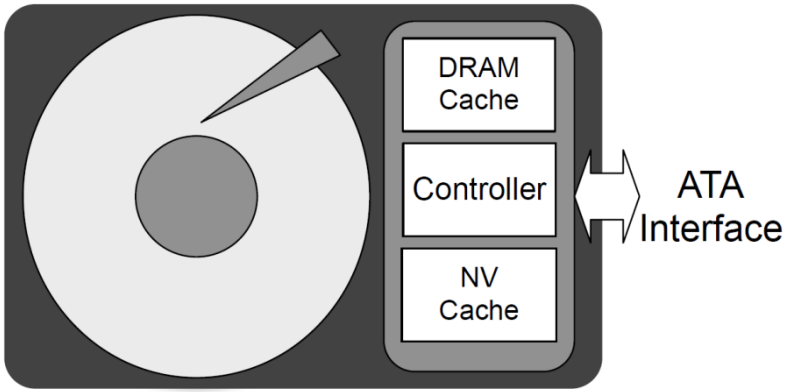
Proceedings of the Fifteenth IEEE International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunications Systems (MASCOTS‘07), 2008

Outline

- Introduction
- Flash-Backed I/O Requests
- Evaluation
- Conclusion

Introduction(1/3)

- One of the biggest bottlenecks in desktop-based computing is the hard disk with I/O write latency being a key contributor
- Hybrid disks place a small amount of flash memory (NVCache) next to the rotating media



	Flash	Rotating Media
Capacity	Smaller	Larger
Random Access	Faster	Slower
Sequential Access	Slower	Faster

Introduction(2/3)

- I/O scheduling algorithms are traditionally implemented to minimize access time to rotating media
- However, with hybrid drives such a presumption may no longer be most efficient
- Improvement
  - Flash-Backed I/O Requests

Introduction(3/3)

- Flash-Backed I/O Requests
  - Augment an I/O scheduler by adding an additional I/O queue in which certain write requests persist in main memory, but are backed in the NVCache
    - Provide additional opportunities for normal I/O requests to be coalesced
    - Read requests a higher chance of being satisfied from main memory

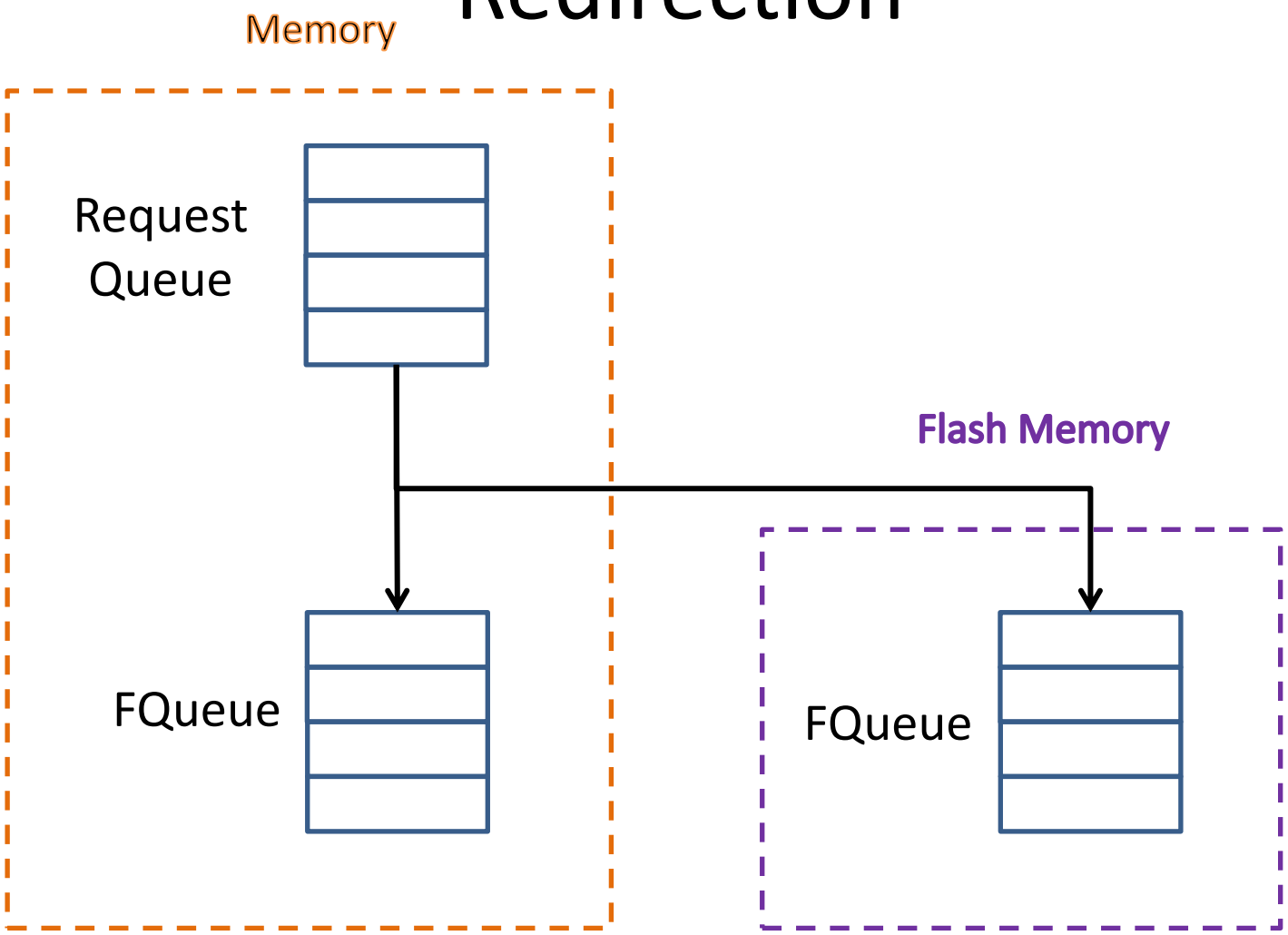
Flash-Backed I/O Requests(1/10)

- I/O Redirection
- When to Redirect
- Idle-Time Processing

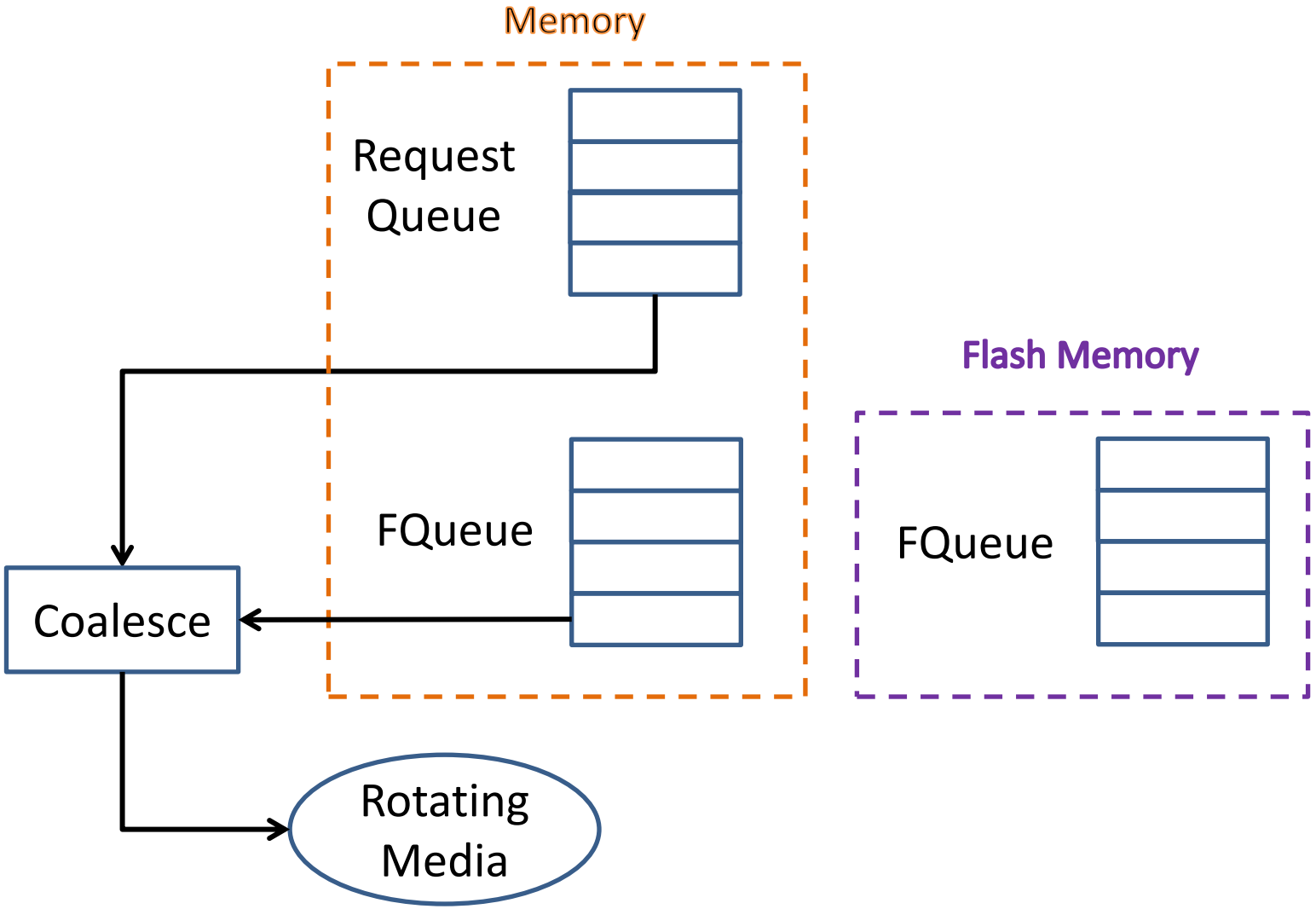
Flash-Backed I/O Requests(2/10)-I/O Redirection

- When a write request is redirected, it is removed from the Request Queue
- Then, the request is added to two new request queues
  - The main memory FQueue
  - A non-volatile FQueue (NVCache)
- The content of the two FQueues at any given time are identical

Flash-Backed I/O Requests(3/10)- Redirection



Flash-Backed I/O Requests(4/10)- Coalesce



Flash-Backed I/O Requests(5/10)- When to Redirect

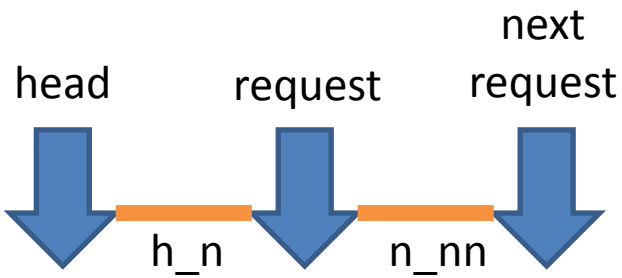
- The goal
  - Result in less disk head seeking
- An algorithm is determined which requests are redirected to the FQueue
- Input
  - Disk drive's head location
  - Current request information
  - The next request information

Flash-Backed I/O Requests(6/10)- Algorithm

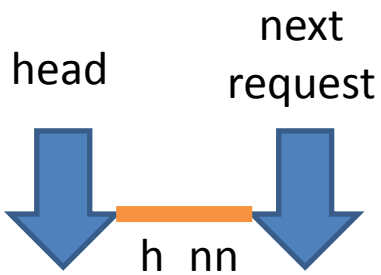
```
/* Decision to redirect request */
REDIRECT_REQUEST(head, request):
    /* Attempt to coalesce request with data from dram fqueue */
    {coalesced, new_req} := COALESCE_WITH_FQUEUE(request)

    if (coalesced == true) then
        /* Remove coalesced data from flash */
        UNPIN_REQUEST_FROM_FLASH(request, new_req)

        /* Submit coalesced request */
        SUBMIT_COALESCED_REQUEST(new_req)

        /* Clear coalesced data from dram fqueue */
        CLEAR_DRAM_FQUEUE_REQUEST(request, new_req)
        return
    endif
```

Flash-Backed I/O Requests(7/10)- Algorithm

```
/*Get I/O access times*/
h_n := ACCESS_TIME(head, request->lbn)
n_nn := ACCESS_TIME(request->lbn, request->next->lbn)
fn := FLASH_ACCESS_TIME(request->size)
h_nn := ACCESS_TIME(head, request->next->lbn)

/*Should redirection occur*/
if ((h_n + n_nn) > (fn + h_nn)) then
    /*pin request in flash*/
    PIN_REQUEST_IN_FLASH(request)

    /*add request to dram fqueue*/
    ADD_REQUEST_TO_DRAM_FQUEUE(request)
Endif

/*submit redirected request*/
SUBMIT_REQUEST(request)
```



V S



Flash-Backed I/O Requests(8/10)-Idle-Time Processing

- By waiting until the Request Queue is empty, and then flushing FQueue requests back to rotating media
- Any request that exists in the FQueue is considered completed

Flash-Backed I/O Requests(9/10)- Flushing Requests

- Flushing requests from the FQueue is comprised of three steps
  - When to flush
  - How much to flush
  - The order of request flushing

Flash-Backed I/O Requests(10/10)- Flushing Requests

- Watermarks (high and low)
  - The high watermark is used to initiate flushing of FQueue requests back to rotating media
  - Requests are flushed from the FQueue until a low watermark is reached
- All I/O operations originating from the Request Queue only execute the COALESCE_WITH_FQUEUE phase

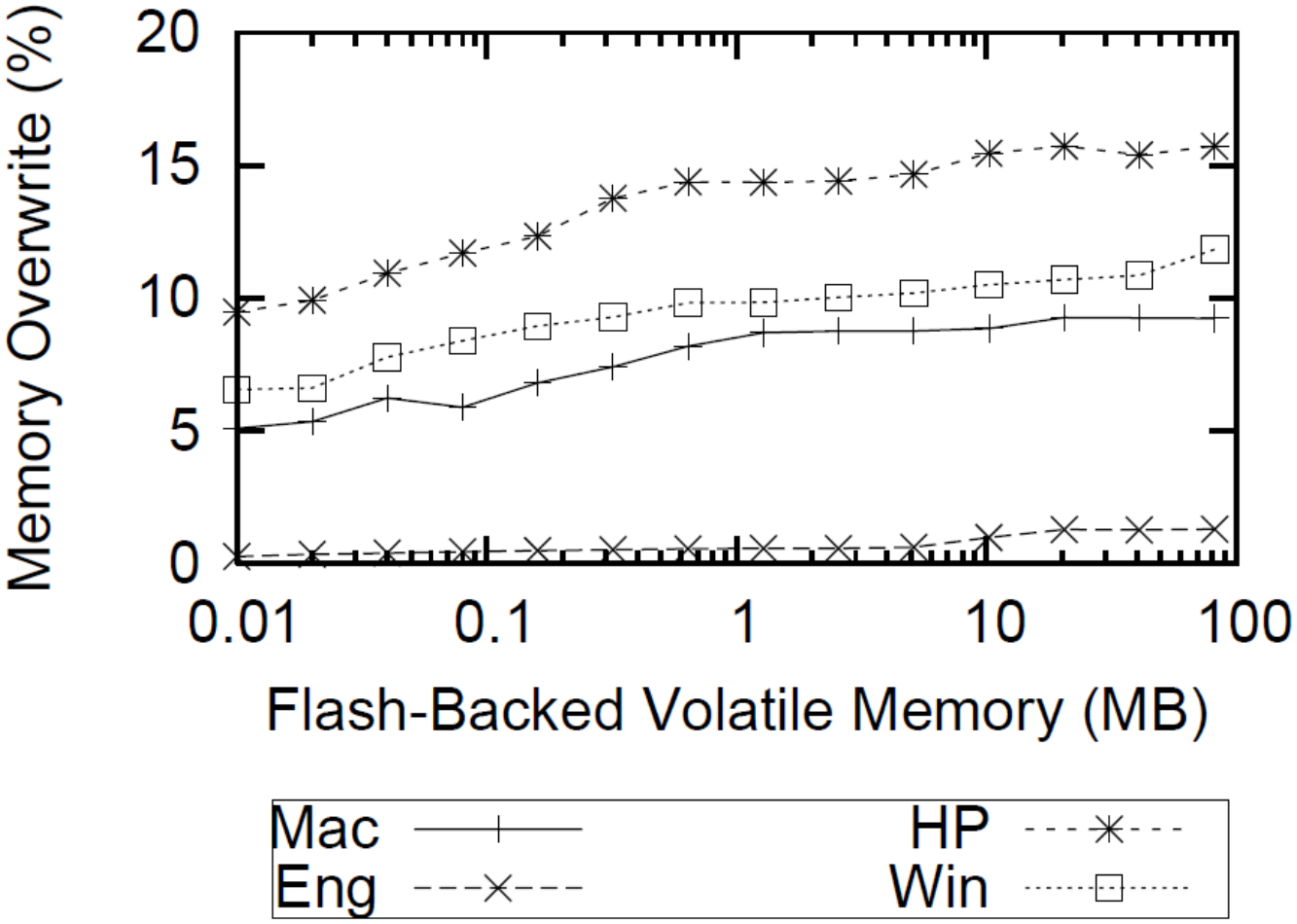
Evaluation(1/3)

- A hybrid disk simulator
  - A Hitachi EK1000 2.5 in drive
  - A Sandisk Ultra II Compact Flash media card

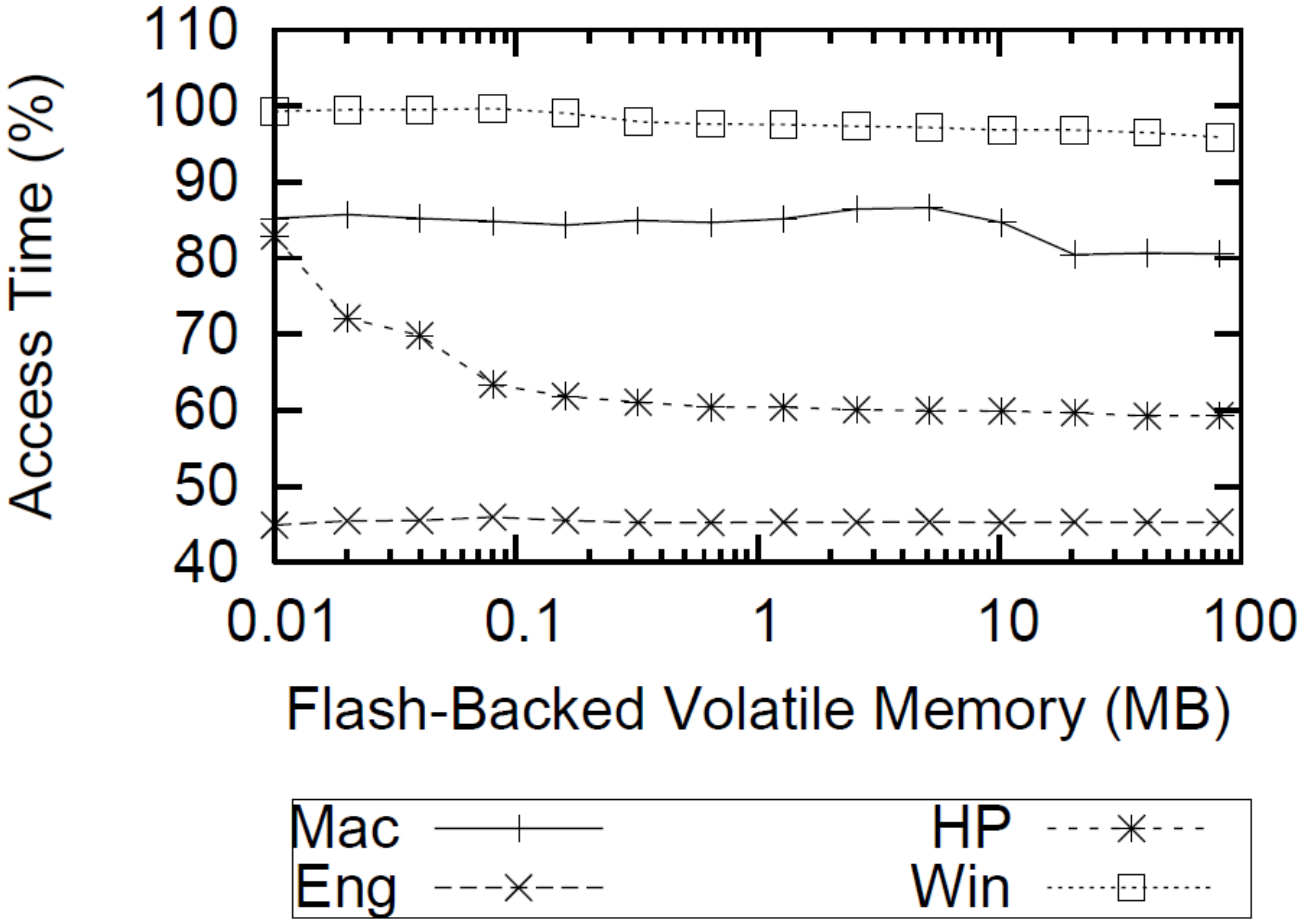
- Workload

Name	Type	Duration	Year
Eng	Linux Engineering Workstation	7 days	2005
HP	HP-UX Engineering Workstation	7 days	1992
WinPC	Windows XP Desktop	7 days	2006
Mac	Mac OS X 10.4 Powerbook	7 days	2006

Evaluation(2/3) -In-memory overwrites



Evaluation(3/3)- Normalized I/O write latency with flash-backed I/O requests



Conclusions

- Disk write latency is a significant component of the overall I/O bottleneck
- Flash-Backed I/O Requests uses the flash memory to reduce write latency by selectively caching write requests to the NVCache