# A Hybrid Disk-Aware Spin-Down Algorithm with I/O Subsystem Support

**Conference Paper** · April 2007

DOI: 10.1109/PCCC.2007.358900 · Source: DBLP

**3 authors:**

Timothy Bisson
University of California, Santa Cruz
**15** PUBLICATIONS **457** CITATIONS
SEE PROFILE

Scott A. Brandt
University of California, Santa Cruz
**192** PUBLICATIONS **6,422** CITATIONS
SEE PROFILE

Darrell D. E. Long
University of California, Santa Cruz
**307** PUBLICATIONS **8,724** CITATIONS
SEE PROFILE

# A Hybrid Disk-Aware Spin-Down Algorithm with I/O Subsystem Support

Timothy Bisson Scott A. Brandt Darrell D.E. Long

*Department of Computer Science*
*University of California, Santa Cruz*

## Abstract

*To offset the significant power demands of hard disk drives in computer systems, drives are typically powered down during idle periods. This saves power, but accelerates duty cycle consumption, leading to earlier drive failure. Hybrid disks with a small amount of non-volatile flash memory (NVCache) are coming on the market. We present four I/O subsystem enhancements that exploit the characteristics of hybrid disks to improve system performance: 1) Artificial Idle Periods, 2) a Read-Miss Cache, 3) Anticipatory Spin-Up, and 4) NVCache Write-Throttling. These enhancements reduce power consumption, duty cycling, NVCache block-erase impact, and the observed spin-up latency of a hybrid disk, resulting in lower power consumption, greater reliability, and faster I/O.*

## 1. Introduction

Hard disks consume a significant amount of power. In general purpose computing, hard disks can be responsible for as much as 30% of a system's power consumption [12, 16]. This percentage will only increase as current CPU trends lean toward increasing the number of cores versus the single core clock rate [11], hard disks use faster rotational speeds, and multiple hard disks per (desktop) system become more prevalent. In large storage systems, hard disks can dominate system power consumption: 86% [1] and 71% [2] of the total power consumption in EMC and Dell storage servers, respectively. As a result, there are several motivations to decrease the power consumed by hard disks, from increasing battery lifetime in mobile systems to reducing financial costs associated with powering and cooling large storage systems.

To reduce hard disk power consumption, spin-down algorithms are used, which put a disk in a low-power mode while it is idle. In a low-power mode, such as standby, the platter is not spinning and the heads are parked, reducing power consumption. Researchers have proposed several spin-down algorithms, which are very efficient at reducing hard disk power consumption [8, 10, 20]. These algorithms are typically time-out driven, spinning down the disk if the time-out expires before a request occurs. Adaptive spin-down algorithms vary the time-out value relative to request inter-arrival times. They are very effective and approach the performance of an optimal off-line algorithm which knows the inter-arrival time of disk requests *a-priori* [13].

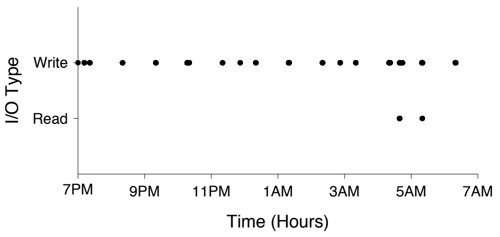


**Figure 1. Disk accesses for 12 hours (7PM - 7AM) on an idle Windows XP system.**

Although spin-down algorithms are effective at reducing hard disk power consumption, pathological workloads can completely negate a spin-down algorithm's power saving benefit, prematurely causing a disk to exceed its duty cycle rating, and significantly increasing aggregate spin-up latency. Such pathological workloads, which periodically write to disk, are not uncommon. Both Windows and UNIX systems exhibit such behavior. For example, Figure 1 shows the periodic disk request pattern of an idle Windows XP system. In UNIX systems, applications such as task schedulers (cron daemon), mail clients, and CUPS (printer service) periodically write to disk.

Upcoming hybrid disks will place a small amount of flash memory (NVCache) logically next to the rotating media [21], as shown in Figure 2. The first hybrid disks will either have 128MB or 256MB of NVCache in a 2.5 in form factor [1]. A host can exploit the NVCache to achieve faster random access and boot time because it has constant access time throughout its block address space as shown in Figure 3, while rotating media suffers from rotational and seek latency. Access time for this particular device is roughly equal to $c + bs \div bs\_off$, where $c$ is a 2.2ms constant overhead, $bs$ is the desired blocksize, and $bs\_off$ is 4KB. In addition to the potential performance increase, hybrid disks can potentially yield longer spin-down durations—the NVCache can service I/O while the disk platter and

[1]Samsung to Unveil First Commercial, Hybrid Hard Drive Prototype for Windows Vista at WinHEC. May 2006

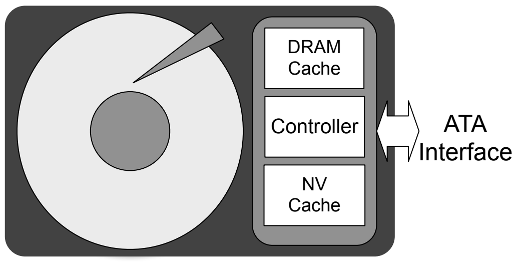


**Figure 2. Hybrid Disk**

arm are at rest, such as the write requests from Figure 1. Note that because flash memory is non-volatile, NVCache-stored data is persistent across power loss.

To exploit the underlying media characteristics of hybrid hard disks for improved power management, we present four enhancements to increase power savings, reliability, and reduce observed spin-up latency: *Artificial Idle Periods* that extend idle periods relative to observed I/O type; a *Read-Miss Cache* that stores NVCache read-miss content in the NVCache itself; *Anticipatory Spin-Up* that spins the rotating media up in anticipation of an I/O operation not serviceable by the NVCache; and, *NVCache Write-Throttling* that limits the reliability impact imposed on the NVCache because of I/O redirection.

## 2. Hybrid Disk Overview

We now present an overview of a hybrid disk and how its NVCache can be managed by a host operating system using a modified set of ATA commands, according to the T13 specification for hybrid disks [19]. The four enhancements are presented in Section 3. Sectors stored in the NVCache are either pinned or unpinned, which when referred to as a collection are known as the pinned and unpinned set, respectively. The host manages the pinned set, while the disk manages the unpinned set.

Hybrid disks will also have a new power mode, *NV Cache Power Mode*, which can be set and unset by the host. In this mode, I/O is directed to the NVCache unpinned set while the disk "aggressively" tries to keep the rotating media spun-down. Defining and implementing "aggressive" is left to the drive vendor's discretion. Although the hybrid disk controls the spin-down policy, the host controls the minimum time rotating media must remain spinning after a spin-up, providing the host with some control over the underlying spin-down algorithm.

The host controls I/O to the NVCache pinned set. Sectors can be pinned in the NVCache, and pinned sectors can be removed or queried. The pinned attribute feature is intended to increase random access performance, although it can also facilitate better power management. The host can flush a specific amount of unpinned content to rotating media to make room for more pinned sectors. However, pinned sectors cannot be evicted to create unpinned space. Addressing multiple sectors at a time (up to 64K sectors) is possible using an extents-based mechanism called LBA

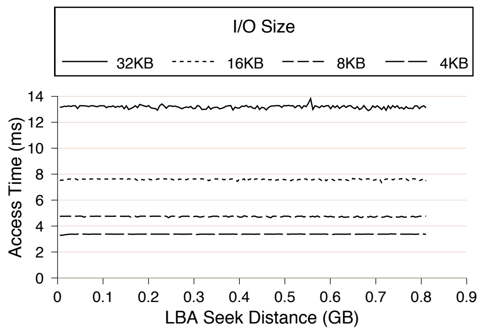


**Figure 3. SanDisk Ultra II 1GB CF Card. Access time with 99% confidence interval.**

Range Entries. A host can also specify the source when adding pinned sectors to the NVCache: host or rotating media, by setting a Populate Immediate bit. This capability gives the host control over NVCache functionality: better random access or spin-down performance.

A host has additional control over a hybrid disk. It can query the disk for spin-up time, read/write NVCache throughput, and the maximum pinnable sectors.

### 2.1. NVCache Utilization

We now discuss the mechanism in which a host can leverage a hybrid disk to provide power management functionality. The host controls rotating media state with traditional power management commands, and NVCache commands to manage the pinned set. Fine-grain spin-down algorithms can be implemented because the host is informed when rotating media power state changes occur. While the rotating media is spun-down, the host should use the NVCache store, query, and read commands to redirect I/O to the NVCache. If the NVCache does not have the requested read data, or it is full before a write, the rotating media must be spun-up, the request satisfied, and NVCache content flushed to disk (while ensuring data coherency) using both pinned set removal and traditional disk I/O commands. The host can put the disk in standby mode again when the spin-down algorithm deems it desirable to do so.

We assume this method because it provides us with complete control over a hybrid disk, allowing us to implement a fine-grain adaptive spin-down algorithm and I/O subsystem enhancements to exploit a hybrid disk's media characteristics. Alternatively, a host could rely on the NV Cache Power Mode to provide all aspects of power management. However, there are several limitations with this approach: the minimum high-power time is not dynamic, it assumes the disk controller implements the correct spin-down policy, and the NVCache may not be a suitable I/O destination for certain workloads. A host could implement its own coarse-grain spin-down algorithm, by repeatedly entering and exiting the NV Cache Power Mode, recording I/O response times to implicitly infer when the rotating media is spun-up. In this way, a host can utilize its own spin-down algorithm, but still has no control over NVCache management.

Note that we omit pinned and unpinned references for the remainder of this work as we no longer refer to the unpinned set.

### 2.2. Hybrid Disk Reliability

Mean Time To Failures (MTTF) and Mean Time Between Failures (MTBF) are widely used metrics to express disk reliability. However, disk manufacturers also provide a duty cycle rating. Duty cycle rating is the number of times rotating media can be spun down before the chances of failure increase to more than 50% on drive spin-up. When controlling a disk's power state with a spin-down algorithm, the duty cycle metric is potentially more important than either MTBF or MTTF because a spin-down algorithm results in an accelerated consumption of duty cycles. In addition to duty cycles, hybrid disks also have flash memory (NVCache) reliability—flash memory blocks have a rated number of erase cycles they can endure before errors are expected.

Today's hard disks generally consume 5–10 times more energy while in active than in standby mode [26]. As a result, adaptive spin-down algorithms are very aggressive—it is more efficient to spin-down after only a few seconds of idle time. With such short idle times the number of duty cycles increases dramatically. Duty cycle terminology varies, depending on drive class and technology. Typically, 3.5 in drives refer to duty cycles as Contact Start/Stop Cycles (CSS), where the head comes to rest on a landing zone on the platter during a power-down. An alternate technology, ramp load/unload, is typically used in notebook drives, where the head comes to rest off the side of the platter. Drives using CSS technology have duty cycle ratings in the range of 50,000, while drives with ramp load/unload technology are in the range of 500,000, mostly due to reduced stiction effects.

With current compact flash specifications, the number of erase operations per block is typically rated at 100,000 [3] with 256KB sized erase blocks [4]. A hybrid disk containing a 256MB NVCache can keep its rotating media spun-down while up to 256MB of data is written to it. With optimal wear-leveling and a write-before-erase architecture, a 256MB device can endure over 100 million erase operations before becoming unerasable. An optimal wear-leveling algorithm spreads all writes across the entire device's physical address space while write-before-erase architecture always writes data corresponding to the same LBA to an empty physical location to ensure data corruption does not occur on a bad overwrite. By exceeding the block erase rating, flash memory blocks may become unerasable, but are still readable. To a host, a hybrid disk with unerasable NVCache blocks should appear as a traditional disk.

## 3. Hybrid Disk-Aware Spin-Down Algorithm with I/O Subsystem Support

Spin-down algorithms which control the power state of traditional hard disks are efficient at reducing disk power consumption. There is little room for improvement of such algorithms, which dynamically adjust to the most power-efficient time-out using machine learning techniques [13]. Hybrid disks present an opportunity for spin-down algorithms to further reduce power consumption while minimizing the performance and reliability impact they impose on the media itself. We now describe four spin-down algorithm and I/O subsystem enhancements.

### 3.1. Artificial Idle Periods

Spin-down algorithms controlling traditional disks compute the idle period as $current\ time - last\ access\ time$, where $last\ access\ time$ is the time of the last disk access. If the idle period exceeds the current time-out, the rotating media is spun-down. I/O type, whether read or write, is ignored because any request, regardless of type, will cause the rotating media to spin-up. This is not true of hybrid disks as one of the intents for adding an NVCache to a hard disk is to extend the duration of spun-down periods by servicing I/O to and from the NVCache. Note this assumes the block I/O layer or disk driver is aware of the rotating media's power state, and will redirect I/O while it is at rest. Our previous work describes a mechanism implemented in the block-layer to redirect I/O to and from a physically separate flash-based NVCache while the disk is spun-down [7]. With such support, NVCache utilization is conveyed to the spin-down algorithm in the context of extended spin-down periods.

A hybrid disk-unaware spin-down algorithm will still ignore I/O type because it believes any I/O will cause a spin-up. However, with the above redirection mechanism, such an assumption is false—write requests are actually unlikely to cause a spin-up. Therefore, we present *Artificial Idle Periods*, a spin-down algorithm modification for a hybrid disk which considers I/O type when computing a disk's idle time, by recording idle time as time since the last *read* request. When a request occurs for a disk in the active mode, the time-out value is reset only on a read request. The idle period is thus artificially increased to $current\ time - last\ read\ access\ time$. As a result, even if a hybrid disk is actively servicing requests, it can be spun-down and remain so, provided I/O consists only of write requests.

Such a modification has several implications. First, duty cycles may be consumed faster; idle periods are artificially increased so a disk will spin-down sooner and probably more frequently. Second, I/O performance may degrade with sequential write workloads as flash sequential write throughput is only a fraction of rotating media, which must still be periodically flushed to disk. Finally, the NVCache will endure more erase-operations resulting from its increased workload, decreasing its expected lifetime.

## 3.2. Read-Miss Cache

As we will show in our evaluation, even with Artificial Idle Periods, typically less than 10% of the NVCache is used per spin-down period to cache writes. Although the NVCache is checked for desired read requests, reads are still typically responsible for initiating spin-ups. NVCache cached writes are not successful at servicing read requests because the host operating system is likely to be idle (rotating media is spun-down) and not evicting buffer cache pages quickly. As a result, most read requests will be satisfied by the buffer cache.

To ensure that read requests are satisfied by the NVCache we propose a *Read-Miss Cache*, an area in the NVCache that is populated with unsatisfied NVCache read requests (read-misses). The hypothesis being that read-misses causing a disk to spin-up are likely to cause a disk to spin-up again. When an NVCache read-miss occurs while the rotating media is spun-down, the requested content is read from the newly spun-up disk and returned to the file system. It is stored in the Read-Miss Cache and subsequent sequential reads are also stored in the Read-Miss Cache, which we refer to as preloading. Preloading stops when a non-sequential read or write request occurs. Only the most frequently preloaded content is stored in the NVCache—preloading data into the NVCache merely updates its frequency count, including the original read-miss.

The Read-Miss Cache size is dynamic. However, a maximum size constraint can be supplied to bound its growth, represented as a percent of the total NVCache. There is also a static minimum Read-Miss Cache size set to 1%. The Read-Miss Cache grows when a read-miss occurs causing the disk to spin-up and shrinks when a write operation cannot be stored in the NVCache because there is no available room. The dynamic size of the Read-Miss Cache is computed below (`rmc_size` represents its current size):

```
spin-up();
if (read-miss && rmc_size < MAX_RMC_SIZE)
    rmc_size += 1;
else if (full && rmc_size > MIN_RMC_SIZE)
    rm_size /= 2;
```

By making the Read-Miss Cache size and amount of data stored on a given spin-up dynamic, we can utilize the NVCache more efficiently while avoiding the effects of quick workload fluctuations.

## 3.3. Anticipatory Spin-Up

Spin-up latency is a significant issue associated with spin-down algorithms because I/O is suspended until the rotating media spins up and becomes ready to service requests again. With an aggressive spin-down algorithm, hours of spin-up latency may be incurred in a single day. Although a disk's spin-up latency is fixed, the observed spin-up latency can be reduced if the disk is spun-up before a request cannot be serviced.

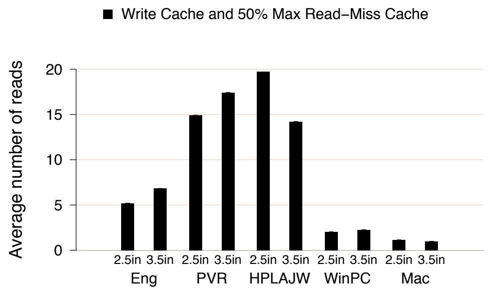


**Figure 4. Average reads from first Read-Miss Cache read to read-miss cache miss**

We propose *Anticipatory Spin-Up*, which attempts to reduce the observed spin-up latency by spinning the rotating media up in anticipation of an I/O operation that cannot be satisfied by the NVCache. Anticipatory spin-up occurs for both write and read requests. To anticipate when the NVCache will become full, we track the current I/O rate, *rate*, at which sectors are written to the NVCache for the current spin-down period. With the number of empty sectors left, we can easily predict when the write-cache will become full. Comparing the predicted time till the NVCache is full with the disk's spin-up latency, we anticipatively spin-up the disk when:

$spin\_up\ latency \leq sectors\_left \div rate.$

Predicting when a read cannot be satisfied is also important because such an I/O is typically associated with a synchronous file system operation. Unfortunately, such a prediction is often difficult because it relies on knowing, *a-priori*, which files and blocks of those files will be read while the disk is spun-down (although we do rely on the LFU Read-Miss Cache to satisfy read requests). We still attempt to use the Read-Miss Cache to perform Anticipatory Spin-Up. As shown in Figure 4, with the Read-Miss Cache on, the average number of read requests satisfied per spin-down period is between 1 and 20 requests. Therefore, we keep a weighted running average, *M*, of Read-Miss Cache I/Os satisfied per spin-down period:

$M = (ops + w \times M_{n-1}) \div (1 + w)$

where *ops* represents the number of I/Os satisfied in the most recent spin-down period. We also keep a weighted running average of the inter-arrival times of subsequent Read-Miss Cache read requests to provide an average rate at which subsequent reads occur to the read-miss cache. Using *M* and the read-request inter-arrival rate, *R*, we spin-up the disk anticipatively when:

$spin\_up\ latency \leq (M - ops) \times R$

While the rotating media is spun-down, *ops* is incremented with each Read-Miss Cache read request. When the disk is spun-up, its final value is used in the weighted average calculation. To prevent *M* from perpetually shrinking we provide *ops* with a boost of $\log(ops)$ when the disk was anticipatively spun-up, but the following I/O was a read which could have been satisfied by the Read-Miss

|  | Compact Flash | Notebook 2.5 in Drive | Desktop 3.5 in Drive |
|---|---|---|---|
| Read/Write | .172W | 2W | 12.6W |
| Seek | - | 2.3W | 13W |
| Performance Idle | - | 1.8W | - |
| Active Idle | - | 1.1W | - |
| Low Power Idle | - | .85W | - |
| Idle | 2.5mW | - | 9.3W |
| Standby | 2.5mW | .2W | .8W |
| Spin-up Time | - | 3 Sec | 15 Sec |
| Erase/Duty Cycles | 100,000 | 600,000 | 50,000 |
| Capacity | 256MB - 8GB | 60GB - 100GB | 500GB |

**Table 1. Sandisk Ultra II CompactFlash Memory Card, Hitachi Travelstar E7K100, and a Hitachi Deskstar 7K500 Specifications**

Cache, or a write which could have been satisfied with write-caching. It is important to note that while the disk is anticipatively spun-up, I/O is still serviced by the NVCache until the entire spin-up latency is endured (unless it cannot service the I/O).

## 3.4. NVCache Write-Throttling

One of the implications of using a flash-based NVCache to cache writes is that its lifetime may be substantially decreased, especially when combined with periodic write workloads and Artificial Idle Periods. To ensure the NVCache's lifetime is predictable, we propose *NVCache Write-Throttling*. Write-throttling regulates the amount of data written to the NVCache while the rotational media is spun-down so that it will last for a specified number of years.

A maximum throughput is set, *Max* MB/S, according the desired NVCache lifetime (such as 100 years). A running NVCache average throughput, $T$, is recorded as:

$T = total\_MB\_written/(current - start)$

where *start* is set to when the spin-down algorithm began using the NVCache, *current* is the current time, and *total MB written* is the total megabytes written to the NVCache thus far. If the running NVCache average throughput exceeds *Max*, the rotating media is not permitted to spin-down. Note that write operations storing sectors in the Read-Miss Cache contribute to *total MB written*.

Therefore, in order for the spin-down algorithm to put the rotating media to rest, the time-out must exceed the idle period, and the running NVCache average throughput must not exceed *Max*. The spin-down algorithm, including Artificial Idle Periods, becomes:

```
artificial_idle_period = current - last_read_access;
if (T < Max && artificial_idle_period > time-out)
    spin_down();
```

## 3.5. Implementation

We implemented hybrid disk functionality in the Linux kernel, mimicking a hybrid disk using flash and a traditional disk by redirecting I/O traffic at the block I/O layer [7]. While the disk is spun-down, I/O is intercepted at the block layer and redirected to flash memory. To evenly spread out block erase cycles and reduce page-remappings, redirected writes are appended to flash in log order. Each redirected request is prepended with a metadata sector describing the original I/O: starting LBA, length, spin-down interval, etc. The redirected LBA numbers are stored in memory (along with the associated flash locations) to speed up read requests to flash. When the flash fills up or a read-miss to it occurs, the corresponding disk is spun-up and the flash sectors are flushed to their respective locations on disk.

We also built a a simulator to model a hybrid disk (power state specifications in Table 1) to allow expedient evaluation of the proposed enhancements with several week-long block-level traces. The simulator considers power relative to the given trace for different power states: read/write, seek, idle, standby, and spin-up. For the notebook drive, power state transitions to and from performance, active, and low-power idle are done internally to the drive and are workload dependent. Therefore, we make a worst case assumption and assume the drive is always in low-power idle, providing a lower bound on spin-down algorithm performance, relative to idle power. Read/write I/O power is computed using the disk's maximum specified I/O rate, and seek power is computed with the drive's average seek time rating for non-sequential I/O.

The spin-down algorithm implemented is the multiple experts spin-down algorithm. It is an adaptive spin-down algorithm developed by Helmbold *et al.* [13]. The spin-down algorithm is based on a machine learning class of algorithms known as Multiple Experts. In the dynamic spin-down algorithm, each expert has a fixed time-out value and weight associated with it. The time-out value used is the weighted average of each expert's weight and time-out value. It is computed at the end of each idle period. After calculating the next time-out value, each expert's weight is decreased proportional to the performance of its time-out value.

## 4. Evaluation

To evaluate the proposed enhancements, we use several different block-level access traces, shown in Table 2. We use four desktop workloads and a personal video recorder workload. Each workload is a trace of disk requests, and every entry is described by: I/O time, sector, sector length, and read or write. The first workload, *Eng*, is a trace from the root disk of a Linux desktop used for software engineering tasks; the ReiserFS file system resides on the root disk. The trace was extracted by instrumenting the disk driver to record all accesses for the root disk to a memory buffer, and transfer it to userland (via a system call) when it became full. A corresponding userland application appended the memory buffer to a file on a separate disk. The trace, *HPLAJW*, is from a single-user HP-UX workstation [22]. The next trace, *PVR*, is from a Windows XP machine used

| Name | Type | Duration | Year |
|---|---|---|---|
| Eng | Linux Engineering Workstation | 7 days | 2005 |
| PVR | Windows XP w/ Beyond TV | 7 days | 2006 |
| HPLAJW | HP-UX Engineering Workstation | 7 days | 1992 |
| WinPC | Windows XP Desktop | 7 days | 2006 |
| Mac | Mac OS X 10.4 Powerbook | 7 days | 2006 |

**Table 2. Block-Level Trace Workloads**

as a Home Theater PC running the personal video recording application, Beyond TV. The *WinPC* trace is from an Windows XP desktop used mostly for web browsing, electronic mail, and Microsoft Office applications. The block-level traces for both Windows systems were extracted using a filter driver. The final trace, *Mac* is from a Macintosh PowerBook running OS X 10.4. The trace was recorded using the Macintosh command line tool, fs_usage, by filtering out file system operations and redirecting disk I/O operations for the root disk to a USB thumb drive.

The physical devices we present results for are a Sandisk Ultra II Compact Flash card, a Hitachi Travelstar E7K100 2.5 in drive, and a Hitachi Deskstar 7K500 3.5 in drive. The power consumption for each state are shown in Table 1. Note that in all figures except Figure 9, we show results using the 2.5 in drive. We present results for both a a 2.5 in and 3.5 in drive in Figure 9 to motivate placing an NVCache in a 3.5 in form factor.

## 4.1. Write Cache and Artificial Idle periods

Figure 5(a) shows the results of making the I/O subsystem aware of a hybrid disk's NVCache, such that a write request occurring while the rotating media is at rest, is redirected to the NVCache. This figure shows the percentage of time the disk can remain spun-down as a function of the NVCache size. The Eng trace benefits the most from the write cache by increasing its spin-down time from 71% to 92%, which translates into an increase of slightly more than one and half days of spin-down time for the seven day trace. This is primarily due to the periodicity of writes-while-idle which occur more frequently than any another workload. The other workloads also benefit from a write-cache by increasing their spun-down time by 4–10%.

Figure 5(b) shows the percentage increase in spun-down time by adding Artificial Idle Periods to write-caching. This plot shows that Artificial Idle Periods significantly increase the percentage of time a disk is spun-down. The most significant benefit comes when the NVCache is less than 1MB. By adding Artificial Idle Periods to NVCache with less than 1MB, its utilization increases as I/O redirection to the NVCache occurs sooner and more frequently. With larger NVCache sizes, Artificial Idle Periods is utilized less often, and so its impact is less pronounced. However, the percentage increase by adding Artificial Idle Periods still stabilizes between 3.5% and 5% for all but the PVR workload, which stabilizes at a 27% increase in spun-down time. The PVR workload benefits from artificial write period so much because its workload consists of periodic write requests without interleaving read requests.

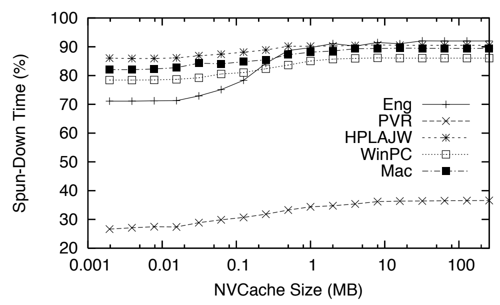
Although write-caching and Artificial Idle Periods are excellent solutions to decrease the time a disk is spent in standby mode, it is important recognize the associated reliability impact. Figure 5(c) shows the number of expected years to elapse before the 2.5 in disk exceeds the duty cycle rating (600,000). By enabling write-caching while the disk is spun-down, reliability increases with respect to utilized NVCache size. As the NVCache exceeds 10MB, reliability stabilizes because it becomes under utilized beyond this point. Figure 5(d) also shows the expected years before the disk will exceed the duty cycle rating, but with Artificial Idle Periods on. In this figure, we see that reliability decreases relative to write-caching alone. A decrease in reliability is expected because the spin-down algorithm has become more aggressive. However, the expected years before exceeding the duty cycle rating is still more than two and half years for the the Mac trace, the lowest of the five workloads. Note that without write-caching or Artificial Idle periods, the Mac workload would exceed the duty cycle rating in seven months.

## 4.2. Read-Miss Cache

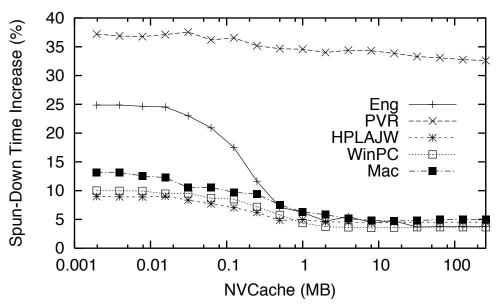
Figure 6 shows the results for a 256MB NVCache with write-caching, Artificial Idle Periods, and a Read-Miss Cache as a function of the maximum Read-Miss Cache size. Figure 6(a) shows the number operations satisfied by the Read-Miss Cache while the rotating media is spun-down. This figure shows that with less than half the NVCache enabled for the Read-Miss Cache, the working set of read requests to the NVCache is captured.

Figure 6(b) shows the average Read-Miss Cache size as a function of the maximum Read-Miss Cache size. The average Read-Miss Cache size grows linearly with respect to the maximum size until 90%, after which it deteriotes quickly, confirming that usually 10% of a 256MB NVCache is used for write-caching and Artificial Idle Periods. The PVR workload is an exception as it stabilizes at 27% because of its high NVCache utilization from television content recording.

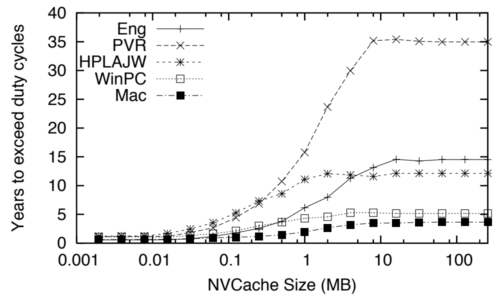
Figure 6(c) shows the percentage increase in spun-down time by adding a Read-Miss Cache to an NVCache with write-caching and Artificial Idle Periods. Here we see that the Read-Miss Cache only increases the spun-down time



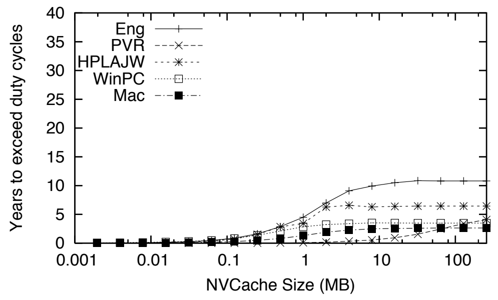
(a) Percent-time spun-down: write-caching only



(b) Adding Artificial Idle Periods to write-caching

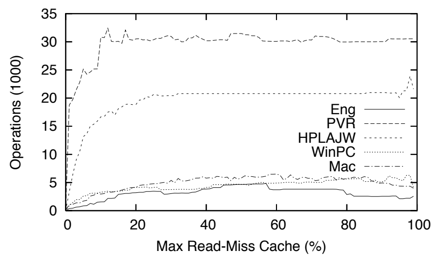


(c) Reliability: write-caching only

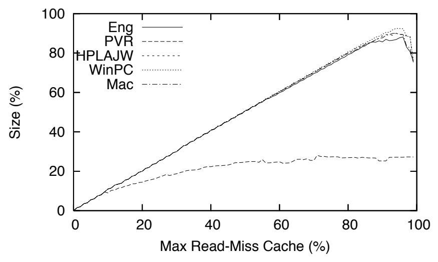


(d) Reliability: write-caching and Artificial Idle Periods

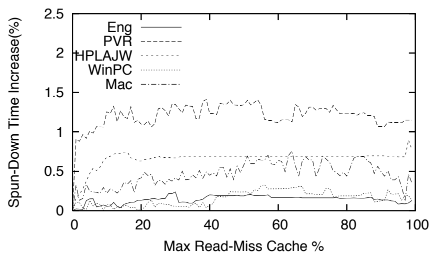
**Figure 5. Write-caching and Artificial Idle Periods**



(a) Read operations satisfied by Read-Miss Cache while spun-down



(b) Average Read-Miss Cache size



(c) Adding Read-Miss Cache to NVCache with write-caching and Artificial Idle Periods

**Figure 6. Read-miss cache: 256MB NVCache with write-caching and Artificial Idle Periods**

percentage by at most 1.5%. Note that with only write-caching and Artificial Idle Periods enabled for a 256MB NVCache, all but the PVR workload are spun-down for 90–95% of the workloads, which leaves little room for improvement. Although the Read-Miss Cache does not increase spun-down time significantly, there is still a consistent increase when the Read-Miss Cache is allowed to use the entire NVCache. This means that its adaptive size prevents it from negatively impacting performance. Additionally, the thousands of read operations it satisfies enables Anticipatory Spin-Up functionality.

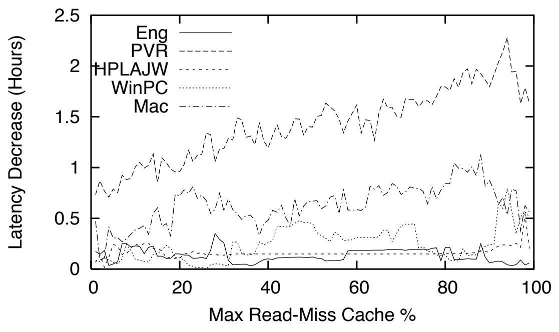
### 4.3. Anticipatory Spin-Up

Figure 7 shows the results for Anticipatory Spin-Up using the metric of observed spin-up latency. Figure 7(a) shows the decrease in latency by adding Anticipatory Spin-Up to a 256MB NVCache with write-caching, a Read-Miss Cache, and Artificial Idle Periods as a function of the maximum Read-Miss Cache size. This graph shows that Anticipatory Spin-Up is capable of predicting when a read cannot be satisfied by the Read-Miss Cache.

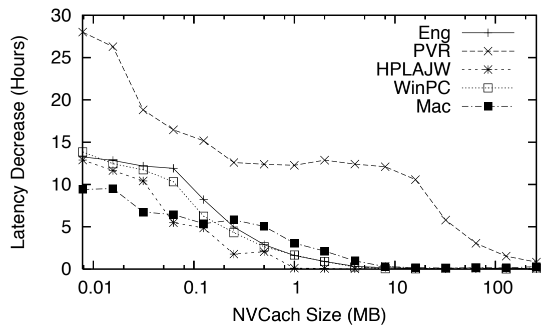
Figure 7(b) shows the decrease in observed latency by adding Anticipatory Spin-Up to an NVCache with write-caching, Artificial Idle Periods, and a 50% maximum Read-Miss Cache as a function of the NVCache size. This figure shows that adding Anticipatory Spin-Up can significantly reduce latency with smaller NVCache sizes because the NVCache becomes full more frequently, which Anticipatory Spin-Up can predict. The notable exception is the PVR workload, which consistently fills even a 256MB NVCache.

### 4.4. NVCache Write Throttling

Figure 8 shows the the effects of NVCache Write-Throttling on rotating media spin-down time, as a function of the desired NVCache lifetime. The block-erase cycle rating is 100,000, and the NVCache has write-caching, a 50% Read-Miss Cache, Artificial Idle Periods, and Antici-



(a) Adding Anticipatory Spin-Up to 256MB NVCache with write-caching Read-Miss Cache, and Artificial Idle Periods



(b) Adding Anticipatory Spin-Up to 256MB NVCache with a write cache, 50% read miss cache, and Artificial Idle Periods

**Figure 7. Anticipatory Spin-up**

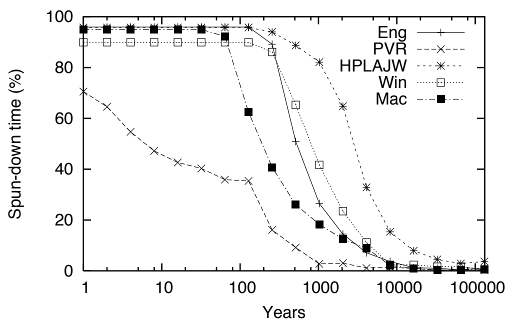


**Figure 8. NVCache Write Throttling**

patory Spin-Up enabled. Once the maximum I/O threshold is exceeded, the percentage of spun-down time deteriorates linearly. All but the PVR workload can be specified with a lifetime of at least 256 years before the throttling begins. Even for the PVR workload, the NVCache isn't throttled until 10 years.

### 4.5. 2.5 in and 3.5 in Hybrid Disks

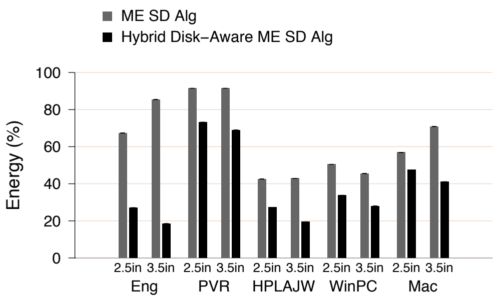
To verify the enhancements' effectiveness, it is important to see how they, as a whole, compare against a traditional spin-down algorithm without I/O support using the metrics described thus far. Additionally, to motivate the inclusion of an NVCache in a 3.5 in form factor we include results for a 3.5 in drive (Hitachi 7K500).

Figure 9 shows percentage of energy consumed, percent time spun-down, latency, and reliability, for both 2.5 in and 3.5 in disks. In these figures, *ME SD* refers to using a traditional spin-down algorithm without underlying media awareness. *Hybrid Disk-Aware ME SD* refers to using a 256MB NVCache with write-caching, Artificial Idle Periods, a 50% maximum Read-Miss Cache, Anticipatory Spin-Up, and 10 year NVCache Write-Throttling.

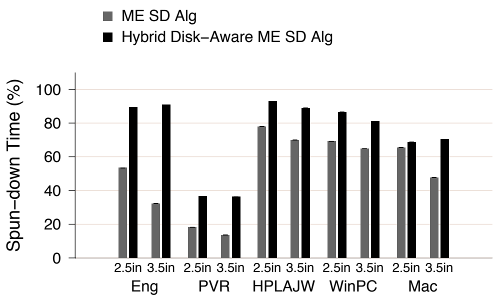
Figure 9(a) shows the percentage of energy consumed relative to the baseline (no spin-down algorithm employed). The energy consumption decrease from ME SD to the Hybrid Disk-Aware ME SD is 10–40 % and 17–67 % for the 2.5 in and 3.5 in disk, respectively. Figure 9(b) shows the increase in spun-down time by adding the described enhancements, which is 25.1–60.1 hours and 38–98 hours, for the 2.5 in and 3.5 in disks, respectively. The best performing workload, Eng, has the biggest decrease in energy and most increase in spun-down time, for both the 2.5 in and 3.5 in disks. The worst workload varies between the HPLAJW, WinPC, and Mac workload.

With respect to latency, Figure 9(c) shows the enhancements reduce the observed latency for the 3.5 in disk by 7.5–46 hours and 6.8–16.2 hours for the 2.5 in disk. It is important to note that for two workloads: Eng and Mac, the latency for the 3.5 in disks is reduced from 48.6 to 2.6 hours and 47 to 6.8 hours, respectively. Regarding reliability, Figure 9(d) shows that the time to exceed the duty cycle rating increases by 2.9–15.7 years and .47–.63 years for the 2.5 in and 3.5 in disks respectively. The 2.5 in disk workloads have such a drastic increase because their duty cycle rating is 600,000, while the 3.5 in disk duty cycle rating is only 50,000. The PVR workload has the biggest increase for both form factors because its workload utilizes the NVCache the most. For all workloads with the 2.5 in form factor, the rated duty cycles aren't exceeded for at least 3 years. For all the workloads with the 3.5 in form factor, the rated duty cycles are always exceeded within a year, except for the PVR workload. However, the enhancements still increase the expected lifetime by several months for each workload.

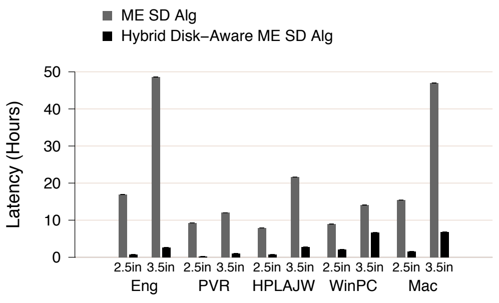
The four metrics: energy consumption, spin-down time, spin-up latency, and duty cycling, are not typically in concert with each other. Decreasing energy consumption and spin-down time typically means a more aggressive spin-down algorithm, which results in an increase in spin-up latency and duty cycling as the rotating media is spun-down more often. However, these results confirm that with the proposed enhancements there is a significant improvement of all four metrics. These results also show that a 3.5 in disk running a spin-down algorithm can also benefit from an NVCache. However, in order not to exceed the duty cycle rating for several years, the spin-down algorithm must be conservative.



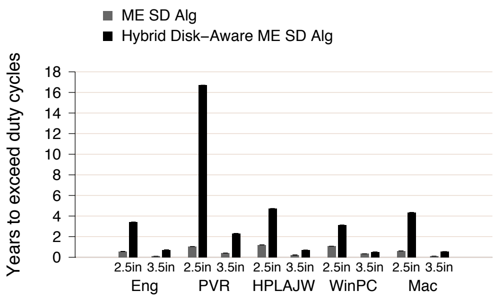
(a) Energy



(b) Percent of total workload disk is spun-down



(c) Observed Aggregate Spin-up Latency



(d) 600K and 50k duty cycle rating for 2.5 in and 3.5 in disk, respectively

**Figure 9. Hybrid Disk-Aware Spin-Down Algorithm (256MB NVCache): write-caching, 50% maximum Read-Miss Cache, Anticipatory Spin-Up, Artificial Idle Periods, and NVCache Write-Throttling**

## 5. Related work

There is previous work motivating the use of a non-volatile cache to increase disk performance [6, 14, 22]. These works generally conclude that a small non-volatile memory write-cache can significantly increase performance by reducing disk traffic. With hybrid disks soon to be available, hybrid disk/non-volatile memory file systems, such as Conquest and Hermes can be evaluated for their effectiveness at increasing file system performance by leveraging on-board non-volatile memory [18, 24].

Previous works have also looked at reducing hard disk power consumption using non-volatile memory. FLASH CACHE proposes to place a small amount of flash directly between main memory and disk, as an additional level in the caching hierarchy to decrease power-savings as well as increase performance [17]. Nvcache focuses completely on reducing power management, and therefore has a completely different architecture [7]. Although Anand *et al.* don't use non-volatile memory, they propose *ghost hints* to anticipatively spin-up a hard disk in a mobile system context while redirecting read I/O to the Internet during disk spin-up [5].

Microsoft proposes to use hybrid disk drives to reduce hard disk power consumption, and decrease boot-time and application launch in their upcoming Microsoft Vista Operating System [21]. They claim a hybrid disk can be spun-down by up to 96% of the time with a 1GB NVCache. Unfortunatelly, neither algorithms nor workloads are described.

Duty cycle is only one metric for disk drive reliability. Disk drive reliability must also factor in duty hours, temperature, workload, and altitude [23]. Mean Time To Failures (MTTF) and Mean Time Between Failures (MTBF) are widely used metrics to express disk drive reliability. However, these metrics must considered with care, as they are often incorrect [9]. IDEMA has proposed a revised MTBF rating based on disk age [15].

Most adaptive spin-down algorithms for traditional disks mention that disk reliability decreases when using a spin-down algorithm, but don't quantitatively describe the impact. Greenawalt modeled the effects of different fixed time-out values and its impact on power conservation and disk reliability [12]. They use a Poisson distribution to simulate inter-arrival access patterns and consider duty cycles as detracting *X* hours from the MTBF rating.

Other strategies to save hard disk power involve pushing power management to applications. Weissel *et al.* [25] propose that energy-aware interfaces should be provided to applications. Such interfaces can be used to convey priority or state information to an operating system. For example, deferrable I/O interfaces can be used by applications to in-form the operating system that particular I/O requests may be differed.

## 6. Conclusions

Hybrid disks augment traditional disks by placing a small amount of flash memory (NVCache) in the drive itself. An operating system can leverage hybrid disks to reduce power consumption by redirecting I/O to the NVCache while the rotating media is at rest, thus reducing spin-up operations and power consumption.

This paper explored four spin-down algorithm and I/O subsystem enhancements which leverage upcoming hybrid disks: Artificial Idle Periods, a Read-Miss Cache, Anticipatory Spin-Up, and NVCache Write Throttling. Artificial Idle Periods reduces hard disk power consumption by observing that write operations will not cause the rotating media in a hybrid disk to spin up. The Read-Miss Cache reduces read-operations that frequently cause spin-up and improves the accuracy of Anticipatory Spin-Up, which minimizes the observed spin-up latency of rotating media. Finally, NVCache Write Throttling ensures expected lifetime of the NVCache.

## References

[1] Symmetrix 3000 and 5000 enterprise storage systems product description guide. http://www.emc.com, 1999.

[2] Dell poweredge 6650 executive summary. http://www.tpc.org/results/individual_results/Dell/dell_66_50_010603_es.pdf, 2003.

[3] Executive summary: Flash quality. http://www.adtron.com/pdf/AdtronFlashQual121103.pdf, 2003.

[4] Sandisk flash memory cards wear leveling. http://www.sandisk.com/Assets/File/OEM/ApplicationNotes/CompactFlash/AppNoteCFHostv1.0.pdf, 2003.

[5] M. Anand, E. B. Nightingale, and J. Flinn. Ghosts in the machine: interfaces for better power management. In *Proc. of the 2nd International Conference on Mobile Systems, Applications, and Services*, pages 23–35, Jun 2004.

[6] M. Baker, S. Asami, E. Deprit, J. Ousterhout, and M. Seltzer. Non-volatile memory for fast, reliable file systems. In *Proc. of the 5th International Conference on Architectural Support for Programming Languages and Operating Systems*, pages 10–22, Oct 1992.

[7] T. Bisson, S. A. Brandt, and D. D. Long. Nvcache: Increasing the effectiveness of spin-down algorithms with caching. In *Proc. of the 14th International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunication Systems*, pages 422–432, Sep 2006.

[8] F. Douglis, P. Krishnan, and B. Bershad. Adaptive disk spin-down policies for mobile computers. In *Proc. of the 2nd Symposium on Mobile and Location-Independent Computing*, pages 130–142, Apr 1996.

[9] J. G. Elerath and S. Shah. Server class disk drives: How reliable are they? In *Proc. of the 50th Annual Reliability and Maintainability Symposium*, pages 151–156, Jan 2004.

[10] Y. G. De Micheli. Adaptive hard disk power management on personal computers. In *IEEE Greate Lakes Symposium on VLSI*, pages 50–53, Mar 1999.

[11] D. Geer. Industry trends: Chip makers turn to multicore processors. *IEEE Computer*, 38(5):11–13, 2005.

[12] P. M. Greenawalt. Modeling power management for hard disks. In *Proc. of the 2nd International Workshop on Modeling, Analysis, and Simulation on Computer and Telecommunication Systems*, pages 62–66, Jan 1994.

[13] D. P. Helmbold, D. D. E. Long, and B. Sherrod. A dynamic disk spin-down technique for mobile computing. In *Proc. of the 2nd international Conference on Mobile Computing and Networking*, Nov 1996.

[14] Y. Hu, T. Nightingale, and Q. Yang. Rapid-cache a reliable and inexpensive write cache for high performance storage systems. *IEEE Trans. Parallel Distrib. Syst.*, 13(3):290–307, 2002.

[15] IDEMA. *R2-98: Specification of hard disk drive reliability*. The International Disk Drive Equipment & Materials Association, 1998.

[16] J. R. Lorch and A. J. Smith. Software strategies for portable computer energy management. *IEEE Personal Communications*, 5(3):60–73, 1998.

[17] B. Marsh, F. Douglis, and P. Krishnan. Flash memory file caching for mobile computers. In *Proc. of the 27th Hawaii Conference on Systems Science*, Jan 1994.

[18] E. L. Miller, S. A. Brandt, and D. D. E. Long. HeRMES: High-performance reliable MRAM-enabled storage. In *Proc. of the 8th Workshop on Hot Topics in Operating Systems*, pages 95–99, May 2001.

[19] N. Obr and F. Shu. Non volatile cache command proposal for ata8-acs. http://www.t13.org, 2005.

[20] J. V. P. Krishnam, P.M. Long. Adaptive disk spin-down via optimal rent-to-buy in probabilistic environments. In *Proc. of the 12th Annual International Conference on Machine Learning*, pages 322–330, Jul 1995.

[21] R. Panabaker. Hybrid hard disk and readydrive technology: Improving performance and power for windows vista mobile pcs. http://www.microsoft.com/whdc/winhec/pres06.mspx, 2006.

[22] C. Ruemmler and J. Wilkes. Unix disk access patterns. In *Proc. of the USENIX 1993 Winter Technical Conference*, pages 405–420, Jan 1993.

[23] N. Talagala and D. Patterson. An analysis of error behaviour in a large storage system. Technical Report UCB/CSD-99-1042, EECS Department, University of California, Berkeley, 1999.

[24] A.-I. Wang, P. L. Reiher, G. J. Popek, and G. H. Kuenning. Conquest: Better performance through a disk/persistent-ram hybrid file system. In *Proc. of the USENIX 2002 Annual Technical Conference*, pages 15–28, Jun 2002.

[25] A. Weissel, B. Beutel, and F. Bellosa. Cooperative i/o: A novel i/o semantics for energy-aware applications. In *Proceedings of the 5thSymposium on Operating Systems Design and Implementation*, pages 117–129, Dec 2002.

[26] J. Zedlewski, S. Sobti, N. Garg, F. Zheng, A. Krishnamurthy, and R. Wang. Modeling hard-disk power consumption. In *In Proc. of the 2nd USENIX Conference on File and Storage Technologies*, pages 217–230, Mar 2003.