On the Trade-Offs among Performance, Energy, and Endurance in a Versatile Hybrid Drive

ZHICHAO LI, Stony Brook University
MING CHEN, Stony Brook University
AMANPREET MUKKER, Stony Brook University
EREZ ZADOK, Stony Brook University

There are trade-offs among performance, energy, and device endurance for storage systems. Designs optimized for one dimension or workload often suffer in another. Therefore, it is important to study the trade-offs so as to enable adaptation to workloads and dimensions. As Flash SSD emerges, hybrid drives are studied more closely. However, hybrids are mainly designed for high throughput, efficient energy consumption, or improving endurance—leaving quantitative study on the trade-offs being unexplored. Past endurance studies also lack a concrete model to help study the trade-offs. Lastly, previous designs are often based on inflexible policies that cannot adapt easily to changing conditions.

We designed and developed *GreenDM*, a versatile hybrid drive that combines Flash-based SSDs with traditional HDDs. The SSD can be used as cache or as primary storage for hot data. We present our endurance model together with GreenDM to study the above trade-offs. GreenDM presents a block interface and requires no modifications to existing software. GreenDM offers tunable parameters to enable the system adapt to many workloads. We have designed, developed, and carefully evaluated GreenDM with a variety of workloads using commodity SSD and HDD drives. We demonstrated the importance of versatility to enable adaptation to various workloads and dimensions.

Categories and Subject Descriptors: H.3.m [**Information Storage and Retrieval**]: Miscellaneous

General Terms: Design, Management, Measurement

Additional Key Words and Phrases: Hybrid drive, Solid-State drive, Trade-Offs, Versatility

ACM Reference Format:
Zhichao Li, Ming Chen, Amanpreet Mukker, and Erez Zadok, 2015. On the trade-offs among performance, energy, and endurance in a versatile hybrid drive. *ACM Trans. Storage* 0, 0, Article 0 (2015), 28 pages.
DOI: http://dx.doi.org/10.1145/2700312

1. INTRODUCTION

The total amount of electronic data stored world-wide is rising exponentially. By 2020, that figure is expected to reach 35 Zetta Bytes [Gantz and Reinsel 2010]. It challenges how fast the storage systems can be to store and fetch the data. Studies show that power consumption in the IT infrastructure is critical [Koomey 2011; Li et al. 2012b], up to 40% power consumption of which comes from storage [Schulz 2007]. Therefore, power consumption has become an important factor influencing storage systems design [Storer et al. 2008; Li et al. 2011b; Verma et al. 2010]. Modern computer components such as CPU, RAM, and disk drives tend to have multiple power

This work was supported in part by NSF awards IIS-1251137 and CNS-1302246.
Author's addresses: Z. Li and M. Chen and A. Mukker and E. Zadok, Computer Science Department, Stony Brook University.

DOI: http://dx.doi.org/10.1145/2700312

states with different operational modes [Li et al. 2011b; Delaluz et al. 2002; Gurumurthi et al. 2003]. Among them, traditional magnetic HDDs achieve the worst *power-proportionality* [Barroso and Hölzle 2009], which states that systems should consume power proportional to the amount of work performed. Moreover, as failures in storage systems become a serious concern [Jiang et al. 2008; Pinheiro et al. 2007], the hardware endurance of storage devices matters as well.

Different storage devices differ in speed, capacity, cost, endurance, and power consumption [R. Freitas 2009]. There are trade-offs among these dimensions in storage systems combining different types of devices. Designs optimized for one dimension or workload often suffer in other dimensions and workloads. Moreover, in prior work, we analyzed the energy and performance profiles of server workloads, such as Web servers, email servers, database servers, and file compression [Kothiyal et al. 2009; Li et al. 2011b]. We discovered large deviations for both performance and energy consumption—as much as 10 times—suggesting that there are significant opportunities to save energy and improve performance. Therefore, it is important to study the trade-offs among these dimensions, and develop highly versatile solution to enable adaptation to different workloads.

With the advent of Flash-based Solid State Drives (SSDs) that are more power and performance efficient than HDDs, many considered SSDs as with a front tier storage cache (e.g., VMware's vFlash [vFlash 2012] and Nimble's CASL [nimble casl 2014]) or as the primary storage to better trade-off performance, purchase cost, and capacity (e.g., Apple's Fusion Drive [fusion drive 2012]; Microsoft's Ready Drive [Panabaker 2006]; Western Digital's Solid State Hybrid Drive (SSHD) [wd sshd 2013]; and Tintri's VMstore [vmstore 2013]. Dell even sells a Compellent Flash Array [dell compellent 2013] that combine two types of SSDs together—Single-Level Cell (SLC) and Multi-Level Cell (MLC)—to achieve the above trade-offs.

Many such approaches often aimed for high performance [Kim et al. 2011; Guerra et al. 2011; tier 2012; Xie and Sun 2008; Chen et al. 2011; Strunk 2012], efficient energy consumption [Guerra et al. 2011; Xie and Sun 2008], or improved endurance [Kim et al. 2011; Xie and Sun 2008]. Therefore, quantitative study on the trade-offs among performance, energy, and endurance is largely unexplored. Studying the trade-offs can help understand the relationship among performance, energy, and endurance, and can also help build storage systems that can adapt to different workloads and dimensions. Moreover, current endurance studies are missing a concrete endurance model and metric to help explore the above trade-offs. In addition, past studies often had designs with fixed or inflexible policies that made it difficult to adapt to different workloads. Moreover, many previous approaches usually rely on simulations and refer to manufacturer's energy and performance specifications for benchmarks, instead of using empirical, real-world experiments.

We designed and implemented a Linux Device Mapper [dm 2012] (DM) target named *GreenDM*, and came up with a concrete endurance model and metric to study the trade-offs. GreenDM has two modes: tiering GreenDM where the SSD is used as primary storage for hot data and caching GreenDM where the SSD is used as cache storage for hot data. GreenDM receives data requests from the hybrid virtual device, and then transparently redirects the resulting requests to the underlying block devices. The DM framework offers additional benefits: it can be used with any target device (e.g., replication, multi-path, encryption, redundancy, and snapshots). The DM framework is highly scalable: one can easily configure the virtual device to use multiple physical devices transparently.

GreenDM separates hot data from cold data based on their access patterns: hot data is stored on the SSD and cold data is stored on the HDD. When cold data becomes hot, GreenDM migrates or prefetches it from the HDD to the SSD; conversely, when

hot data becomes cold or more space is needed for hotter data, GreenDM migrates or evicts colder data from the SSD to the HDD.

By utilizing the SSD for hot data before using the HDD, GreenDM improves performance and reduces energy use—as SSDs are typically faster and consume less energy than HDDs. To improve concurrency, GreenDM decouples the data movement between the SSD and the HDD. By keeping mostly cold data on the HDD, GreenDM can spin down the HDD at times and help reduce whole-system energy consumption. By counting the number of physical SSD reads and writes [M. Jung and M. Kandemir 2013; P. Desnoyers 2013] and the HDD start-stop cycles [Guenter et al. 2011], GreenDM tracks vital parameters that impact the endurance of the underlying storage devices. To enable adaptation to different workloads, GreenDM supports several versatile configuration parameters to determine the migration thresholds. These parameters can be tuned in accordance with the workloads and dimensions.

We have evaluated GreenDM with several workloads. We experimented with several configurable GreenDM parameters, analyzed the results, and demonstrated their impact on the trade-offs among performance, energy, and endurance. We also showed the importance of matching configuration parameters to specific workloads to tune the above trade-offs. In the FIU online trace workload, for example, we showed that tiering GreenDM achieved a higher throughput (58–142%) than Mylinear, but consumed more power (4–8%) and further reduced the SSD's endurance by 11–15% under certain conditions. We observed similar trends for caching GreenDM as well. We also demonstrated the importance of matching tunable parameters to different workloads to achieve better trade-offs among performance, energy, and endurance. For tiering GreenDM, a larger extent size (ES) lead to higher throughput and larger energy savings, but reduced the SSD's endurance further; for caching GreenDM, a too large extent size lead to lower throughput and lower energy savings, but it wore out the SSD faster.

Our contributions are four-fold: (1) we developed a device endurance metric; (2) we studied empirically the trade-offs among performance, energy, and device endurance for our hybrid drive (both tiering and caching modes); (3) we compared empirically the tiering and caching modes of the hybrid drive in several dimensions; and (4) we offered a versatile solution to enable adaptation to different workloads and dimensions for the system.

2. DESIGN AND IMPLEMENTATION

We describe GreenDM's design and implementation in this section. Section 2.1 presents the design goals. Section 2.2 shows the system architecture. Section 2.3 details the design. Section 2.4 describes our power-management techniques. Section 2.5 describes the endurance model used for the trade-off study. Section 2.6 presents the implementation details.

2.1. Design Goals

The work was motivated by several concerns in storage systems. With the advent of SSDs, there are now more opportunities to investigate these concerns. Specifically, with GreenDM, our design goals were as follows:

(1) **Hybrid Drive:** we want to build a hybrid drive with efficient data management and additional power management as the benchmark system, where the SSD is used as either a primary storage or as a cache device.
(2) **Trade-offs Study:** we would like to come up with a per-device endurance metric to help study the trade-offs.

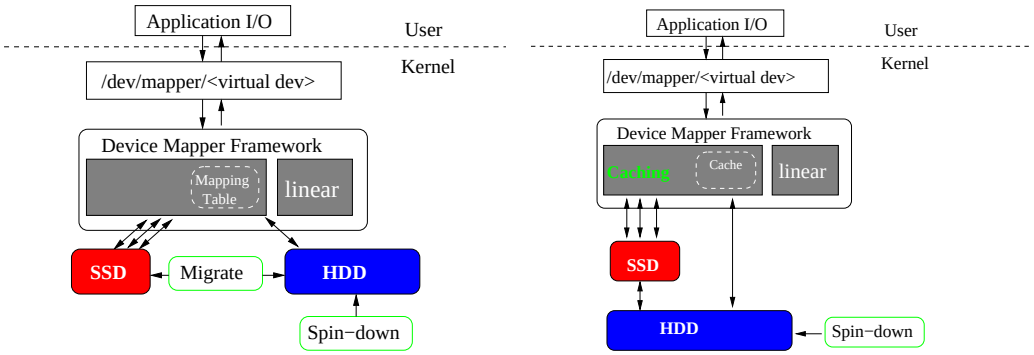
(3) **Versatility:** we want to have versatile policies so that the system can adapt to different workloads.

To build the hybrid drive, GreenDM (1) migrates/prefetches hot data to the SSD, and migrates/evicts cold data to the HDD—useful in workloads that exhibit hot/cold I/O patterns; (2) decouples the data movement between the SSD and the HDD to improve concurrency between CPU and I/O; (3) optimizes the data management by serving I/O requests directly from RAM instead of the SSD whenever possible; (4) throttles data movement between the SSD and the HDD to control the overhead and effectiveness of migrations; (5) uses the lower-power SSD over the HDD and spins down the HDD when it is idle for a sufficient amount of time; and (6) is implemented in the Linux DM framework to be scalable. Note that in this work we did not primarily aim to design for superior performance, highly efficient energy consumption, or a large improvement to the device's endurance. Instead, the aforementioned techniques were intended for the purpose of building a hybrid drive so we could quantitatively study the trade-offs among performance, energy, and endurance.

To help study the trade-offs among performance, energy, and endurance, GreenDM counts and utilizes the number of SSD reads and writes and the number of HDD start-stop (spin-up/down) cycles to estimate the devices' endurance. SSDs especially can wear out quickly and become less durable [Soundararajan et al. 2010], and a mechanical disk drive can only be spun down and up for a limited number of cycles [Joukov and Sipek 2008].

To achieve the versatility goal, GreenDM supports several controllable parameters so that the system can be tuned to different workloads.

2.2. Architecture

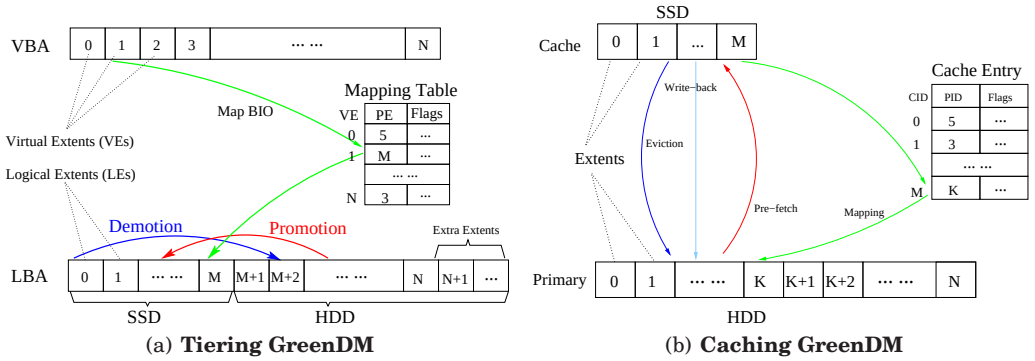


(a) **Tiering GreenDM** (b) **Caching GreenDM**

Fig. 1. **GreenDM Architecture**. *The gray shaded areas are DM targets, usually implemented as loadable kernel modules.*

We implemented GreenDM in the Linux DM framework to benefit from its scalability and flexibility. Figure 1 shows our system's architecture. GreenDM has two modes: a tiering mode (Figure 1(a)) and a caching mode (Figure 1(b)). We detail GreenDM's internals in later sections. *Linear* is another existing DM target that linearly maps from the virtual storage address space to the physical one. GreenDM is scalable: it can be easily configured to use multiple drives with minor code change. However, to better study and understand the fundamental behavior of our hybrid drive, we used a two-drive setup in this paper: one SSD drive and one HDD drive.

2.3. Data Management



(a) **Tiering GreenDM** (b) **Caching GreenDM**

Fig. 2. **Data Management in GreenDM**. *The green arrows show the address translation work-flow from one Virtual / Cache Extent (VE / CE) to one Logical Extent (LE).*

GreenDM tries to keep hot data in the SSD so that the system benefits most from the SSD's superior performance and efficient energy consumption. To achieve this, GreenDM moves hotter data to the SSD, and moves colder data to the HDD as the working set changes over time. To guarantee the correctness of moving data around, tiering GreenDM uses a mapping table and caching GreenDM uses a cache entry to keep track of data movement. Figure 2 illustrates GreenDM's data management. Tiering GreenDM divides the Virtual Block Address (VBA) space and the Logical Block Address (LBA) space into *extents* that are multiple of the (4KB) page size for efficient data management. Caching GreenDM divides the space of the cache device and the primary device into extents as well. The Extent Size (ES) is a configurable parameter, but once configured, the size is fixed for the lifetime of the DM instance. Our extents are atomic units of data movement. Data management involves moving extents between the SSD and the HDD, and then updating the mapping table or the cache entry accordingly.

Mapping table. The mapping table is a core data structure in tiering GreenDM, as shown in Figure 2(a). It has four fields: LE ID, State, Usage Counter, and Time-stamp of the latest access. The LE ID identifies one LE. State represents the accessing state of each extent. The usage counter represents the number of total accesses. The time-stamp records the latest access of one specific extent. The tiering GreenDM populates the mapping table lazily. With a new virtual drive, the table starts empty. The tiering GreenDM creates the mappings as the workload requires it. Compared to fully initializing the table with linear mapping, this approach provides more flexibility to data migration, especially when the workload is light. The tiering GreenDM uses a bit in the State field of each table entry to indicate if the entry is empty or not. The tiering GreenDM uses a bitmap to indicate whether the LEs on both drives are occupied or not. The mapping table and the bitmap together comprise the metadata of tiering GreenDM. Whenever a new VE is accessed, the tiering GreenDM first allocates one free LE and then sets the corresponding mapping entry and the bitmap field properly. To locate a free LE, the tiering GreenDM always starts from the lower LBAs so that it improves the SSD's utilization. To accelerate this operation, the tiering GreenDM maintains an in-memory only free list for free LEs on the SSD.

Cache entry. The caching GreenDM uses a cache entry table instead, to help with data management, as shown in Figure 2(b). The cache entry table maintains mapping information only from the cache device to the lower-level device, and contains not only the four fields in the mapping table of the tiering GreenDM (i.e., extent ID, state, usage counter, and time-stamp of the latest access), but also a dirty flag to indicate whether a cached extent is updated or not. The cache entry table together with the bitmap comprise the metadata of the caching GreenDM.

Data separation. GreenDM separates hot I/Os from cold I/Os based on their access frequencies, and stores them separately to best utilize the tiering hybrid device. Temporal locality suggests that once an extent is accessed, it is likely that the extent will be accessed again soon. In the tiering GreenDM, active I/Os are served through the primary drive (i.e., SSD) and the mapping is established accordingly. Inactive I/Os are kept on the secondary drive (HDD). In the caching GreenDM, active I/Os are first served through the cache drive (SSD) and the cache entry table is updated accordingly. GreenDM is designed this way so that hot I/Os can be served mainly by the fast but smaller SSD, and cold I/Os are held on the slow but larger HDD drive.

Data promotion and prefetching. The tiering and caching GreenDM modes use the same method to move hot data. We name the hot data moving process *promotion* and *pre-fetch* in the tiering and caching GreenDM modes, respectively. Therefore, as an example, we focus here on illustrating the data promotion process of the tiering GreenDM. The same technique is used for the caching GreenDM data prefetching. To speed future accesses, promotions move hot LEs from the HDD to the SSD as the workload changes over time. To detect hot LEs on the HDD, GreenDM counts the number of I/O misses for every LE. An I/O is considered missed when the mapped LE resides on the HDD. An LE is considered hot if the number of I/O misses exceeds the Promotion Threshold (PT). The tiering GreenDM increases the LE miss count if two adjacent I/O accesses to the LE are within a Time Window (TW). Otherwise, the miss count is reset. The PT and the TW are both configurable parameters. Once GreenDM decides to promote a hot LE, it allocates a free LE on the SSD and enqueues the job to a promotion queue. The tiering GreenDM invokes a worker thread to keep dequeuing promotion jobs and copy data from source LEs to destination LEs synchronously. When the mapped LE of one VE is being promoted, accesses to the VE are suspended before being served; then the Tiering GreenDM updates the mapping table. The tiering GreenDM cancels promotion attempts under any of the following conditions: (1) the SSD is full, because promotion requires free space in the SSD; (2) the metadata is being flushed to disk, because promotion has to update the metadata; (3) the Maximum Concurrent Migration Limit (MCML) is reached, because we throttle migration; or (4) there is concurrent access on the extent that is to be promoted, because the extent is already being accessed. Thus, instead of accessing the SSD, the HDD is accessed. This may delay the HDD's spin-down and help maintain SSD's endurance, but increase access latencies.

Data demotion and eviction. The tiering and caching GreenDM modes use similar method to move cold data. We name the cold data moving process *demotion* and *eviction* in the tiering and caching GreenDM, respectively. Therefore, as an example, we focus on illustrating the data eviction process of tiering GreenDM in this paragraph. A similar technique is used for caching GreenDM data prefetching. Demotion moves cold LEs from the SSD to free LEs on the HDD. There are different ways to perform data demotion. One approach is to evict SSD LEs instantly when promotions are taking place but there are no free extents on the SSD. This approach adapts well to the workload changes. However, it can prolong the promotion I/O latencies, which is undesirable. Another alternative is to schedule demotion as a periodic background job. However,

this strategy is detrimental to energy efficiency because it has to wake up the HDD periodically. Instead, Tiering GreenDM schedules demotion in the background when the number of free LEs on the SSD drops to a configurable Low Threshold (LT). Once demotion is launched, it keeps demoting extents until the number of free LEs on the SSD reaches the configurable High Threshold (HT). The default value of HT is higher than the LT so that cold LEs are demoted efficiently in batch without constantly disturbing the HDD. When all LEs are mapped, the tiering GreenDM uses a small number of extra reserved extents in the HDD, as shown in Figure 2(a), to allow the demotion to find free LEs. Otherwise, data migration stalls if no free LE is found. The demotion thread uses the WSClock algorithm to find cold extents and updates the mapping table accordingly. Tiering GreenDM uses a device-mapper kernel thread called `dm_kcopyd`, which copies data between disk drives asynchronously. There are two differences between tiering and caching GreenDM to move cold data. First, caching GreenDM looks up the cache entry table to decide the physical location for the eviction extent, instead of randomly finding one free extents in the HDD. Second, caching GreenDM does not need to reserve extra extents in the HDD for eviction to succeed, as it is guaranteed to map an extent from the SSD to the HDD.

Data movement throttling. GreenDM throttles data movement to improve throughput. Both the mapping table and the cache entry table have one field to count the number of accesses for each Logical Extent (LE). When the Promotion Threshold (PT) of one LE is reached, data promotion/eviction is attempted. The PT is configurable: (1) a larger PT can decrease the number of promotions/prefetchings and reduce the overhead, especially when there are lots of accesses; and (2) when the benefit of one promotion/prefetching exceeds the overhead, a larger PT reduces the potential benefits. Migration is also throttled by the Maximum Concurrent Migration Limit (MCML). The MCML specifies the maximum concurrent promotions/prefetchings and demotions/evictions. The MCML is tunable: (1) a larger MCML value can promote/prefetch hot I/Os to the SSD earlier and prepare free SSD extent slots earlier to benefit future accesses; and (2) a larger MCML value can potentially choke the system as ongoing data movements can freeze other I/O requests. Demotion/Eviction tries to maintain [LT, HT] free extents in the SSD so that promotion/prefetching can just use the free extent instead of waiting for demotion/eviction to proceed. Demotion/Eviction is designed to decouple from promotion/prefetching to improve interleaving between CPU and I/O.

Read/Write Policy. In a tiering system, since the SSD is used as primary storage, reads and writes access the data from the current location either on the SSD or HDD according to the mapping table. Cold data migrates to the HDD and hot data eventually migrates to the SSD using kernel threads. In a caching system, reads and writes access data from the SSD if the data is still there, else from the HDD. If it is an SSD write hit or if there is a write to any of the I/Os that are served from RAM, the system stores information of the pending write-back I/O in a queue, and an asynchronous write-back kernel thread wakes up to flush dirty writes from the SSD to the HDD. To control the rate of the write-back process, our caching system removes duplicated, queued write-back I/Os and caps the maximum queue size. The queue is not blocking the whole system because our experiments show that the queue is around $1/3$ full under pressure test. The current policy can help illustrate the negative effects of the caching write back policy compared with the tiering system that requires no write back at all. The implementation detail of the caching's write policy is based on the eviction process for easy data management. I/O access can be slow down during write-back activity.

Table I. GreenDM Parameters and Abbreviations

Abbrev.	Name	Ex. Values
ES	Extent Size (in 4K units)	4K, 16K, 64K
PT	Promotion Threshold	1, 2, 4, 8
MCML	Max Concurrent Migration Limit	2, 4, 8, 16, 64
SP	Spin Down Policy	On, Off
LT	Low Threshold of demotion/eviction	32, 64
HT	High Threshold of demotion/eviction	64, 128
TW	Time Window length (sec)	30, 60

Serving directly from RAM. To save I/Os, GreenDM serves buffered I/O requests directly from RAM instead of the SSD in case of a successful promotion/prefetching. The size of the RAM buffer is equal to the size of the hot LE. When a hot LE is being promoted/prefetched, I/Os mapped to it will be pending before being served. A naïve approach to serve the pending I/Os is to first move one LE from the HDD to the SSD, and then access the SSD again to serve the pending I/O requests one by one. However, this approach triggers more SSD accesses than needed. Instead, GreenDM first reads the LE data from the HDD to RAM; then, GreenDM serves pending I/Os directly from RAM, before the LE data is written to the SSD. GreenDM invokes the DM API `bio_endio` to indicate that the I/O request was served. For each pending I/O, this saves one SSD I/O cycle by serving directly from RAM. This approach can save many SSD I/Os because when a LE is hot, it is likely to be accessed many times even during the short period of promotion/prefetching. If there are more I/Os accessing the same LE while data is being flushed from RAM to SSD, GreenDM suspends these I/Os in the queue and serves them from the SSD as usual.

Versatility. To enable adaptation to different workloads, GreenDM supports several configurable system parameters: ES, PT, MCML, SP (spin-down policy), LT, HT, and TW. Table I summarizes the parameters in more detail. All parameters can be set at the user level. The ES can be set in the GreenDM configuration file before the tiering hybrid drive starts service. The users can set the remaining parameters at any time by accessing the corresponding Linux `debugfs` entries.

2.4. Power Management

In addition to the above data-management techniques, our GreenDM manages the power consumption of the system to save energy. First, GreenDM saves energy simply by using the SSD in preference to the HDD. To further save power, when the HDD is idle, GreenDM spins down the drive [Zhu et al. 2005]. The side effect of this spin-down is two-fold: (1) it takes time for a spun-down disk to spin back up, and (2) it reduces the HDD endurance if the HDD is spun up and down too frequently as each (mechanical) HDD has a limited number of start-stop cycles. GreenDM spins down the disk when it is idle for at least five seconds, configured by `hdparm`. We chose five seconds because it is the time it takes to spin down the HDD we used. The smaller the time-out latency is, the more aggressive the HDD spin-down policy is. When there is access on the spun-down disk, it spins up automatically.

2.5. Endurance Model

Table II. Devices Wear-out Limits

	Limits
SSD	36,500 GB writes
HDD	300,000 spin up/down cycles

This subsection describes the endurance model we used for the trade-off study.

GreenDM explores the endurance model for both the SSD and the HDD. For the HDD, GreenDM utilizes the number of start-stop cycles as the major factor towards endurance.

For the Flash-based SSD, it suffers from the endurance problem because Flash device requires one block erasure operation before the block can be rewritten. There are different levels of failure modes [P. Desnoyers 2013; Kadav et al. 2009; M. Jung and M. Kandemir 2013]. In failure mode I, the ECC can successfully correct the device bit errors. It happens when the device is not erased beyond the specified maximum value. In failure mode II, the bit error rate goes beyond the ECC's correction capability, but the device is still within the retention period. This can happen when the device is erased beyond the specified maximum value. In failure mode III, even the device goes beyond the retention period. In this work, we examine the SSD endurance within failure mode I.

An SSD's endurance depends on many internal (often proprietary) parameters, some of which are hard or impossible to measure: internal write-amplification factor, wear-leveling techniques, FTL's effectiveness, garbage collection algorithms, reserved space, internal caching, and more. In this paper, we do not attempt to measure these internals. Instead, to help estimate the SSD's endurance, we used 4KB as the default SSD page size, counted each page access (read and write) to the SSD, and formalized our endurance model to study the trade-offs among performance, energy, and endurance.

Moreover, as the real-time endurance relies heavily on the history usage of the devices, GreenDM utilizes delta endurance metrics for both SSD and HDD to show the endurance reduction of each device in any configured experiment. We summarize the endurance models as follows:

$$Endu_{ssd}(t) = 1 - \frac{writes(t)}{Limit_{ssd}}(t > 0) \tag{1}$$

$$Endu_{hdd}(t) = 1 - \frac{\#startstop(t)}{Limit_{hdd}}(t > 0) \tag{2}$$

$$\Delta Endu_{ssd}(\Delta t) = \frac{\Delta writes}{Limit_{ssd}}(\Delta t > 0) \tag{3}$$

$$\Delta Endu_{hdd}(\Delta t) = \frac{\Delta \#startstop}{Limit_{hdd}}(\Delta t > 0) \tag{4}$$

$$0 \leq Endu_{ssd}(t), Endu_{hdd}(t) \leq 1(t > 0) \tag{5}$$

$Endu_{ssd}(t)$ and $Endu_{hdd}(t)$ represent the endurance metric of the SSD device and the HDD device, at time t, respectively. $\Delta Endu_{ssd}(\Delta t)$ and $\Delta Endu_{hdd}(\Delta t)$ represent the delta endurance (i.e., the endurance reduction) of the SSD device and the HDD device during the time period Δt, respectively. The endurance of the SSD at time t is represented by 1 minus the fraction of writes performed at time t (i.e., $writes(t)$) and the total writes limit (i.e., $Limit_{ssd}$). The more writes are performed, the less durable the SSD is. Note that reads also affect the SSD's endurance because erase operations can happen once read disturbance correction kicks in [M. Jung and M. Kandemir 2013]. Since this is fairly recent reported result and there is no quantitative study on the endurance effects of the read disturbance, in our work, we convert the effect of reads to writes based on several certain configurable ratios (e.g., endurance effects caused by reads is calculated by $reads/10$ and $reads/100$). The endurance of the HDD at time t is represented by 1 minus the fraction of start-stop cycles performed at time t (i.e., $\#startstop(t)$) and the total cycles limit (i.e., $Limit_{hdd}$). The more the device performs

start-stop actions, the less durable the HDD is, and the closer it is to failing. We show the limits for both SSD and HDD in table II based on the vendor data-sheet.

To simplify the understanding and use of our endurance metric, we define eu as the unit for the endurance models as shown in Equation 5. We define endurance on a scale of one million parts. The higher the value is, the more durable the device is: a value of 1,000,000 is given to brand new drive that is unlikely to break under failure mode I, and a value of 0 is given to a drive that is almost certain to break under failure mode I in the very near term. For example, a reduction of a device's endurance by $1,000eu$ means that the probability of a device's failure has increased by $\frac{1,000}{1,000,000}$ or 0.1%.

2.6. Implementation Details

Concurrency control. The Linux DM framework supports concurrent block accesses. Since data movement is performed in the back-end, it is possible that data movement and an incoming I/O compete for the same extent. GreenDM uses a spin-lock to protect critical resources, and creates one atomic counter for each extent to ensure that before GreenDM moves data, all I/O requests on associated extents are completed. This counter is incremented once per access on the extent, and is decremented for each I/O request that is finished. If GreenDM observes that the counter of one specific extent is larger than zero, it drops the data movement attempt. If the incoming I/O happens to be in the extent that data promotion/prefetching is going to be performed, GreenDM delays the I/O by putting it into a queue and serve it later. If the incoming I/O happens to be in the extent that data demotion/eviction is going to be performed, GreenDM cancels the data demotion/eviction.

Metadata management. Metadata (e.g., mapping table and bitmap for tiering GreenDM and cache entry table and bitmap for caching GreenDM) is critical for a data-movement based approach. GreenDM stores metadata in RAM for frequent accesses. In case of a power outage, the system may be inconsistent and lose persistent data. Therefore, GreenDM periodically flushes metadata to the SSD for recovery. GreenDM also replicates metadata on the HDD for redundancy. In case of failures, GreenDM reads the latest metadata checkpoint from one of the persistent drives into RAM. We discuss limitations of the in-memory metadata management in Section 5.

Statistics export. To better analyze the dynamic mappings of block I/Os, the effectiveness of data movements, and the status of the running system, GreenDM exports several kernel-space statistics to user space. We implemented such an interface through Linux's `debugfs` with the support of the `seq_file` interface. GreenDM places all debugfs files and directories for GreenDM under `green_debugfs_root`. GreenDM creates a debugfs entry named "stats" to collect statistic information of the running system (e.g., the SSD hit ratio, the number of promotions/prefetchings and demotions/evictions, the system status, etc). GreenDM creates a debugfs entry named "table" to export the mapping table or the cache entry table to user level. These statistics were helpful during the development and analysis phases.

Development cost. We spent two years on this project. We developed around 3,500 LoC in kernel space for the tiering GreenDM, and wrote around additional 500 LoC in kernel space for caching GreenDM. We developed fewer than 100 LoC to add statistics counters for Linear. We used Auto-pilot [Wright et al. 2005] to help benchmarking, but further developed an additional 2,000 LoC in Bash and Python to assist in benchmarking and analysis. To automate raw data parsing and plotting, we developed another 2,000 LoC in Bash and Python. To help profiling the performance, we developed another 1,000 LoC in Python. We also developed around 500 LoC in C++ to replay the traces.

Table III. Trace Workloads Summary

Workload	Drive Size	Reads		Writes	
		Total	Avg Sz	Total	Avg Sz
Web-search	32GB	1,055,236	16KB	212	8KB
FIU online	8GB	655,526	8KB	4,211,786	4KB

3. EVALUATION

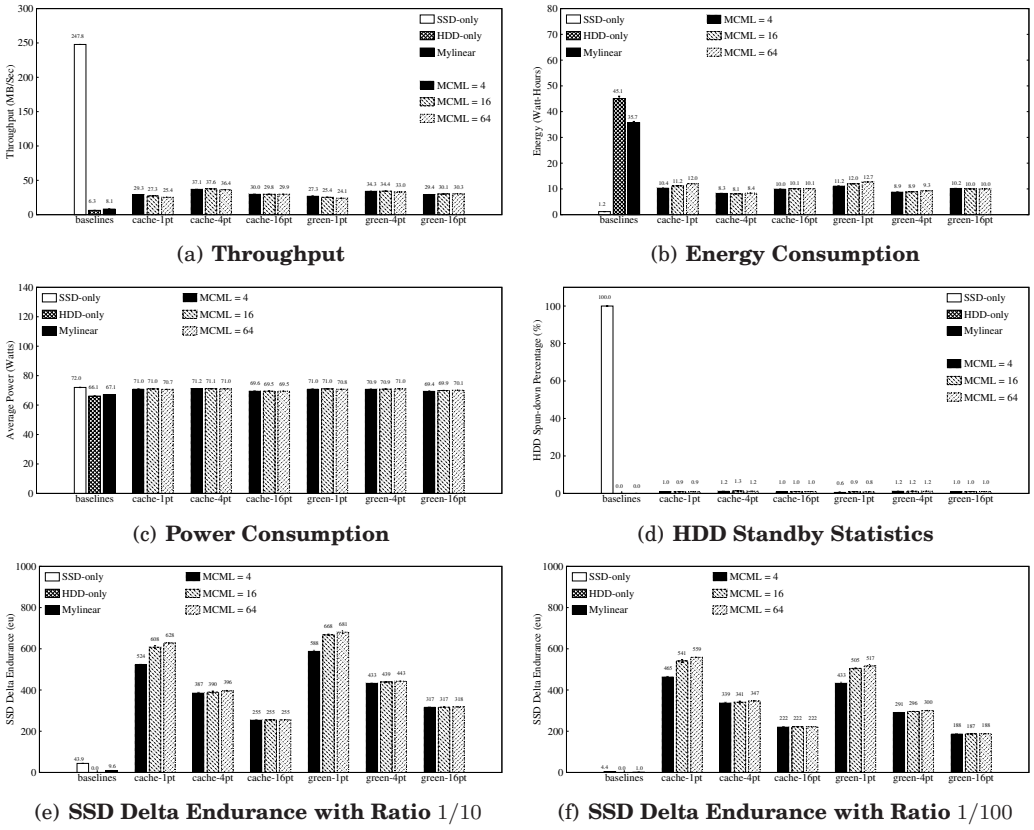
Our evaluation aim to answer the following questions:

(1) What are the tiering and caching GreenDM performance, energy, power, and endurance results compared with other baselines under various workloads?
(2) What are the trade-offs among performance, energy, power, and endurance?
(3) How much do different tunable parameters affect the trade-offs among performance, energy, and endurance under various workloads?
(4) How much does the capacity ratio of the SSD as a fraction over the total capacity affect both tiering and caching GreenDM in terms of throughput and device endurance?

3.1. Experimental Setup

We experimented on two identical Lenovo® ThinkCenter computers. Using lmbench [Brown and Seltzer 1997], we verified that the performance difference of the two machines was within 2%; and that the power consumption difference was within 1.6%. Each server has 4GB RAM and one Intel® Core™ 2 Quad 2.66GHz CPU. We configured the BIOS identically on both machines. As energy consumption is important to our study, we used the default "ondemand" CPUFREQ governor [Zhu and Mueller 2004] and the default "menu" CPUIDLE governor [Li and Belay 2007]. We kept all CPU cores online by default. Our tiering hybrid drive consists of an Intel SSDSA2CW300G3 300GB SSD and a Seagate ST32000641AS 2TB HDD. We used only the middle portion of the HDD's LBA space to average out any ZCAV [zcav 2014] effects. The OS, using a Linux 3.5.0 kernel, ran on a separate SATA drive. We prepared several baselines: (1) SSD-only drive; (2) HDD-only drive; and (3) a linear tiering hybrid drive (i.e., *Linear*) that linearly maps from the VBA space to the LBA space. We added a few statistics counters to Linear and named it *Mylinear* in our experiments. We set the DM `split_io` option so that I/Os are split based on the Extent Size (ES). We used $1/4$ as the default ratio for the SSD partition size out of the total drive size for our hybrid drive. The ratio is just one example for us to study the trade-offs among performance, energy, and endurance for the hybrid drive; it also keeps the SSD size relatively small compared with the workload's working set size. Note that for the caching GreenDM, since the SSD capacity is not counted toward the total capacity, the HDD capacity has to be expanded to build the same amount of total capacity. We further used $1/8$ in Section 3.6, as the new SSD capacity ratio over the total capacity, to show different effects.

We connected each computer to a WattsUP Pro ES in-line power meter [Wattsup 2010], which measures the energy drawn by a device plugged into the meter's receptacle. The power meter uses non-volatile memory to store measurements every second. Its resolution is 0.1 Watt-hours (1 Watt-hour = 3,600 Joules) for energy measurements. The accuracy is $\pm 1.5\%$ of the measured value plus a constant error of ± 0.3 Watt-hours. Its resolution for power measurements is 0.1 Watts. We used the `wattsup` Linux utility to download the recorded data from the meter over a USB interface to the test machine.



(a) **Throughput** (b) **Energy Consumption**

(c) **Power Consumption** (d) **HDD Standby Statistics**

(e) **SSD Delta Endurance with Ratio** 1/10 (f) **SSD Delta Endurance with Ratio** 1/100

Fig. 3. **Web-Search Trace Replay Results**. We configured ES to 1MB in GreenDM, to help with sequential prefetching. HDD spin-down was enabled for all.

3.2. Benchmarks

We evaluated GreenDM (both tiering and caching) carefully with three general purpose workloads: (1) Web-search trace workload; (2) FIU's online trace workload; and (3) File-server workload. We used the Web-search and FIU online trace workloads from the UMass Trace Repository [UMassTraceRepo 2009] and the FIU Trace Repository [FIUTraceRepo 2011], respectively. We summarized these traces' parameters in Table III. Note that the drive sizes are sized to meet the storage requirements of the two workloads, respectively. We used the File-server workload from Filebench [Filebench 2011].

GreenDM's effectiveness depends on the degree of data locality the workload exhibits. Therefore, for the File-server workload, we varied the frequency that files are accessed using Filebench's Gamma distribution [Wilson 2008; gamma distribution 2014].

In this paper, to understand our GreenDM's behavior under different conditions, we focused on parameters that tend to have more impact on the trade-offs among performance, energy, and endurance of the tiering hybrid drive. Thus, for example, we tuned ES, PT, and MCML values for different workloads while keeping the default values for other parameters (i.e., TW=60, LT=64, HT=128). Specifically, we varied: (1) the Promotion Threshold (PT) and the Maximum Concurrent Migration Limit (MCML)

values for the Web-search trace workload; (2) the Extent Size (ES) for the FIU online trace workload; and (3) the MCML and the Gamma values for the File-server workload. To reduce side-effects due to the SSD's Garbage Collection (GC), we issued the TRIM [M. Wei and L M. Grupp and F. E. Spada and S. Swanson 2011] command to the SSD before each experiment. The results we show are general to illustrate the effects of tuning parameters.

We ran all tests a minimum of three times unless otherwise noted. We computed the standard deviations and presented as error bars in figures. we discuss and explain the larger standard deviations below. We used Autopilot [Wright et al. 2005] to automate the benchmarks.

3.3. Web-Search Trace Workload

We replayed the UMass Web-search trace with our own tool in synchronous mode, without introducing any delay between two consecutive I/O requests. Since the trace is block-level, we disabled the OS buffer cache in this experiment. To meet the storage requirement, we set up the tiering and caching hybrids with 32GB storage (i.e., 8GB SSD storage). We scanned the device initially to fill the mapping table such that it could represent a more realistic situation where the mapping table was not initially empty. We present the results in Figure 3. To avoid repeated discussions, we describe the tiering GreenDM as an example.

Figures 3(a), 3(b), and 3(c) show that: (1) the SSD-only drive achieves the highest throughput, the lowest energy consumption, and the highest power consumption; (2) the HDD-only drive achieves the lowest throughput, the highest energy consumption, and the lowest power consumption; (3) tiering hybrids achieve throughput, energy and power consumption in the middle; and (4) among tiering hybrids, various GreenDM configurations achieve higher throughput, lower energy consumption, and higher power consumption compared with Mylinear since the real-world Web-search trace exhibits many hot and cold I/O patterns for GreenDM to manage. Figure 3(d) shows that the HDD is rarely spun down for all benchmarks when the disk spin-down in enabled. The reason is that this workload exhibits high randomness and therefore keeps the HDD active most of the time. Thus, the incurred reduction to the HDD's endurance can be ignored. Figures 3(e) and 3(f) show that: (1) the HDD-only drive does not reduce the SSD's endurance since no I/O accesses the SSD; (2) the Mylinear tiering hybrid drive wears out the SSD the slowest since part of the I/Os goes to the HDD; (3) the GreenDM tiering hybrid drive wears out the SSD the fastest since data migration causes lots of SSD reads and writes; and (4) the SSD-only drive reduces the SSD endurance in the middle since there is no data migration at all.

We now focus our study on the trade-offs for tiering hybrid drives.

Higher throughputs lead to larger energy savings, as shown in Figures 3(a) and 3(b). The reason is that it takes less time to finish the same amount of work when the throughput is higher and the system-level average power consumption between GreenDM and Mylinear is close (see Figure 3(c)).

There are trade-offs between performance and power consumption. As shown in Figures 3(a) and 3(c), GreenDM achieves higher throughput (198–325%) than Mylinear, but consumes slightly more power (5%) since the faster SSD I/Os indirectly keep the CPU and RAM busier, and shift the bottleneck a bit towards the CPU. This keeps the system more active during the run, and shows the trade-off relationship between performance and power consumption for this workload. Note that the SSD-only based system consumes a little bit higher power (1%) than GreenDM because it makes the CPU and RAM even busier.

There are trade-offs between performance and the SSD endurance. As shown in Figures 3(a), 3(e), and 3(f), GreenDM achieves higher performance than Mylinear, but

reduces the SSD's endurance more. When the ratio of read-to-write effect is $1/10$, the reduction goes from 32× to 70× more. When the ratio of read-to-write effect is $1/100$, the reduction goes from 186× to 516× more. The reason is two-fold: (1) GreenDM performs many data migrations to separate hot and cold data; and (2) Web-search workload has many more reads than writes and reads are not as effective as writes in reducing the SSD's endurance [M. Jung and M. Kandemir 2013]. Note that MCML values become less effective when the PT value becomes larger since a larger PT value leads to smaller promotions.

Different tiering GreenDM tunable parameters have different effects on performance, energy, and device endurance. As shown in Figures 3(a), 3(b), 3(d), 3(e), and 3(f), different MCML and PT values affects the performance, energy, and endurance in different ways. For example, when the PT and MCML values are 1 and 64, respectively, GreenDM improves throughput by 198%, saves energy by 64%, and reduces the SSD's endurance by 70× and 516× more when the ratio of read-to-write effect is $1/10$ and $1/100$, respectively. However, when the PT and MCML values are 4 and 16, respectively, GreenDM improves throughput by 325%, saves energy by 75%, and reduces the SSD's endurance by 44× and 295× more when the ratio of read-to-write effect is $1/10$ and $1/100$, respectively. The reason is that different GreenDM parameters yield different benefits and (CPU and I/O) overhead. Medium PT and MCML values tend to achieve the best balance of benefits vs. overhead for this workload: (1) a too large PT value can reduce the migration benefits and a too small PT value can increase the migration overhead (see Figure 3(a)) for this workload; (2) when the PT value is small, large MCML values can consume more CPU and I/O resources on the system; and (3) when the PT value is large, large MCML values can promote hot I/Os to the SSD faster. However, when the PT value is larger, it incurs less reduction to the SSD's endurance (see Figures 3(e) and 3(f)). In sum, there is no single best configuration for this workload. Therefore, to achieve different trade-off goals, the MCML and PT values have to be chosen carefully.

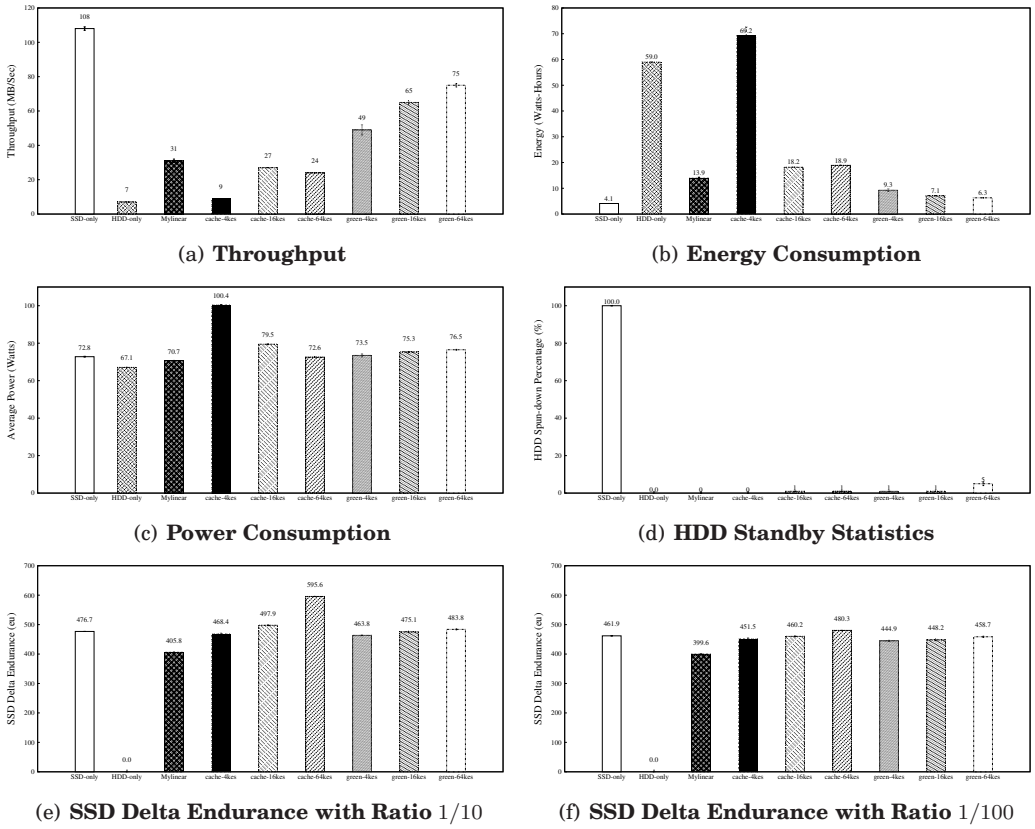
Next, we discuss the results for caching GreenDM. As shown in Figure 3, we can observe similar trade-offs among performance, energy, power, and endurance, and similar versatility effects as we discussed above for the tiering GreenDM. We then compare caching GreenDM against tiering GreenDM in more details. For this Web-search trace workload, caching GreenDM achieves slightly higher throughput (i.e., 4–9%) than tiering GreenDM does when the Pre-fetching Threshold (PT) is 4 and 16; and achieves very similar throughput than tiering GreenDM does when PT is 64, as we can see in Figure 3(a). For the purpose of explanation, the Web-search trace workload has much more reads than writes, as shown in Table III. That means the overhead of the write-back is not going to be significant since there are only a few writes. Moreover, as the primary storage (i.e., SSD) in tiering GreenDM contains either cold or hot data before hand, this can incur additional overhead to the overall throughput. However, the caching device in caching GreenDM only contains hot data. That means the overall throughput of caching GreenDM may be higher than that of tiering GreenDM in some degree if the primary storage initially contains cold data in tiering GreenDM.

The caching GreenDM also consumes similar amounts of energy as the tiering GreenDM does when the PT is 1 and 16. When the PT is 4, the caching GreenDM consumes slightly less energy (i.e., 7–10%) than tiering GreenDM does, as shown in Figure 3(b). The main reason is that the energy consumption is coupled with the throughput since the total amount of workload is the same. The caching GreenDM consumes a similar amount of power as the tiering GreenDM does, as shown in Figure 3(c). The caching GreenDM also rarely spins down the HDD, as shown in Figure 3(d).

Finally, the caching GreenDM also wears out the SSD less (i.e., 8–20%) than that of tiering GreenDM when the ratio of read-to-write effect is $1/10$, as seen in Figure 3(e).

The reason is due to the aggregated primary SSD I/Os in the tiering GreenDM. The caching GreenDM wears out the SSD faster (i.e., 8–19%) than tiering GreenDM does when the ratio of read-to-write effect is $1/100$, as seen in Figure 3(f). The reason is two-fold: (1) this Web-search is read-dominated; and (2) reads do not wear out the SSD as much as writes.

3.4. FIU Online Trace Workload



(a) **Throughput** (b) **Energy Consumption**

(c) **Power Consumption** (d) **HDD Standby Statistics**

(e) **SSD Delta Endurance with Ratio** $1/10$ (f) **SSD Delta Endurance with Ratio** $1/100$

Fig. 4. **Online Trace Replay Results**. As example, we set MCML to 16 and PT to 1. Disk spin-down was enabled for all.

We replayed the FIU online trace using our own tool as mentioned in Section 3.3. We disabled the OS buffer cache as the trace is a block-level one. To meet the storage requirement, we set up the tiering and caching hybrids with 8GB storage (2GB SSD storage). We also scanned the device initially to fill the mapping table. We present the results in Figure 4. To avoid repeated discussions, we use the tiering GreenDM as an example.

Figures 4(a), 4(b), and 4(c) show that: (1) the SSD-only drive achieves the highest throughput, the lowest energy consumption, and a medium power consumption; (2) the HDD-only drive achieves the lowest throughput, the highest energy consumption, and the lowest power consumption; (3) tiering hybrids achieve throughput and energy consumption in the middle; and (4) among tiering hybrids, various GreenDM con-

figurations achieve higher throughput, lower energy consumption, and higher power consumption compared with Mylinear, due to GreenDM's efficient data management. Figure 4(d) shows that GreenDM spins down the HDD to some degree when the ES varies. Otherwise, the HDD is rarely spun down for other benchmarks. Since only a few start-stop cycles are caused, the reduction to the HDD's endurance can be ignored. Keeping the HDD idle could save some power, but since the SSD indirectly helps the CPU stay busier and the spin down/up process consumes more power, the GreenDM system-level power consumption is thus slightly higher than all others. Figures 4(e) and 4(f) show that: (1) the HDD-only drive does not reduce the SSD's endurance; (2) the SSD-only drive wears out the SSD to a moderate degree compared with tiering hybrids; and (3) GreenDM configurations wear out the SSD faster than Mylinear. Next we discuss the trade-offs for tiering hybrid drives.

Higher throughputs lead to lower energy consumption, as shown in Figures 4(a) and 4(b). The reason is similar to what we have explained in Section 3.3. There are trade-offs between performance and power consumption. As shown in Figure 4(c), GreenDM achieves higher throughput (58–142%) than Mylinear. However, it consumes more system-level power on average than Mylinear does, ranging from 4–8%, due to the aforementioned reasons. There are trade-offs between performance and the SSD endurance. As shown in Figures 4(a), 4(e), and 4(f), GreenDM achieves higher throughput than Mylinear does, but it reduces the SSD's endurance by 14–19% more and by 11–15% more when the ratio of read-to-write effect is $1/10$ and $1/100$, respectively, as explained in Section 3.3.

Different tiering GreenDM Extent Sizes (ES) have different effects on GreenDM's performance, energy consumption, and device endurance. As shown in Figures 4(a), 4(b), 4(d), 4(e), and 4(f), different ES values lead to different results. For example, when the ES is 4KB, GreenDM improves throughput by 58%, saves energy by 33%, and reduces the SSD's endurance by 14% and 11% more when the ratio of read-to-write effect is $1/10$ and $1/100$, respectively. However, when the ES is 64KB, GreenDM improves throughput by 142%, saves energy by 55%, and reduces the SSD's endurance by 19% and 15% more when the ratio of read-to-write effect is $1/10$ and $1/100$, respectively. The larger the ES is, the more effective the sequential pre-fetching algorithm is. Therefore, it leads to higher throughput and larger energy savings. However, larger ES causes more reduction to the SSD's endurance. It suggests the ES has to be chosen carefully for the system to achieve the best trade-offs because there is no single optimal configuration.

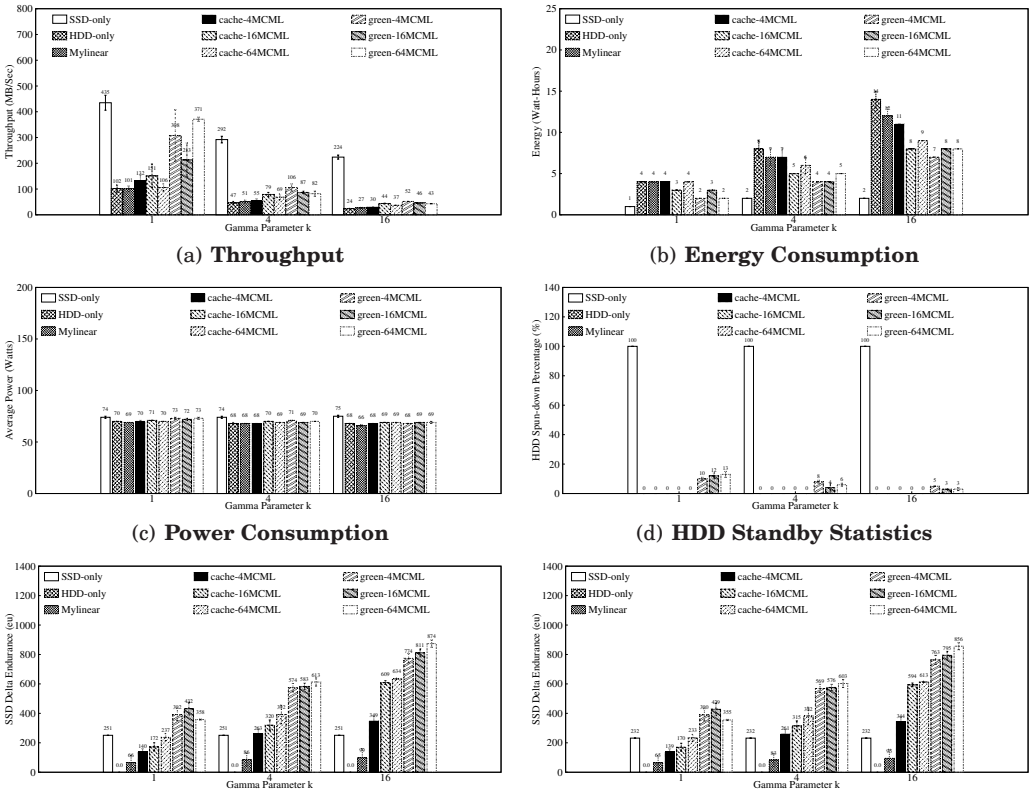
We next discuss the results for the caching GreenDM. As we can see in Figure 4, there are similar trade-offs among performance, energy, power, and device endurance and similar versatility effects as we discussed above for the tiering GreenDM. We then compare caching GreenDM against tiering GreenDM in more details. For this FIU online trace workload, caching GreenDM achieves less throughput (i.e., 58–82%) than tiering GreenDM does when the ES varies, as we can see in Figure 4(a). For the purpose of explanation, the online trace workload has lots of writes, as shown in Table III. That means the overhead of the write-back can be a bottleneck when it comes to throughput. A medium ES in achieves the best throughput in caching GreenDM because it achieves the best balance between prefetching benefit and write-back overhead.

Caching GreenDM also consumes more energy (i.e., 2–6$\times$) than tiering GreenDM does, as shown in Figure 4(b). The reason is that the total energy consumption is coupled with the throughput and the total amount of workload is the same. Caching GreenDM consumes similar power as tiering GreenDM does, as seen in Figure 4(c). The only big difference is that for caching GreenDM, when the ES is 4K, caching GreenDM consumes much higher power than other conditions. The reason is that

when the ES is 4K, there are much more write-back I/Os that cause the system (i.e., CPU, RAM, and I/O system) to be even more active. The caching system also rarely spins down the HDD, as shown in Figure 3(d).

Lastly, caching GreenDM also wears out the SSD faster (i.e., 1–23% and 2–5%) than the tiering GreenDM does, when the ratio of read-to-write effect is $1/10$ and $1/100$; this can be seen in Figure 4(e) and Figure 4(f), respectively. The main reason is that caching GreenDM has more write-back I/Os than the aggregated primary SSD I/Os in the tiering GreenDM. Therefore, the caching GreenDM can wear out the SSD faster than tiering GreenDM does.

3.5. File-Server Workload



(a) **Throughput** (b) **Energy Consumption**

(c) **Power Consumption** (d) **HDD Standby Statistics**

(e) **SSD Delta Endurance with Ratio** $1/10$ (f) **SSD Delta Endurance with Ratio** $1/100$

Fig. 5. **Fileserver Workload Results**. We configured the ES to be 128KB in GreenDM. It is equal to the average I/O size to avoid the migration waste and I/O split overhead. The PT value was fixed at 1, as example. Disk spin-down was enabled for all configurations.

We ran the File-server workload with a Gamma distribution in Filebench [Filebench 2011]. We varied the Gamma value to show different results. The smaller the Gamma value is, the higher the data locality is since smaller Gamma values lead to narrower file accesses: that is, a certain subset of data items (i.e., Logical Blocks) would be referenced more than others. We configured the usable RAM size to be 1GB to ensure that the workload would generate many low-level I/Os. To meet the storage requirement,

we set up the tiering and caching hybrids with 8GB (2GB SSD storage). We report the results in Figure 5. We first focus on the tiering GreenDM to avoid repeated discussions. Note that since the OS buffer cache is enabled, to better estimate the SSD endurance change, we assume fully low-level I/Os for SSD-only benchmark.

Figures 5(a), 5(b), and 5(c) show that: (1) the SSD-only drive achieves the highest throughput, the lowest energy consumption, and the highest power consumption; (2) the HDD-only drive achieves the lowest throughput, the highest energy consumption, and medium power consumption; (3) tiering hybrids achieve throughput, energy and power consumption in the middle in general; and (4) among tiering hybrids, various GreenDM configurations achieve higher throughput, lower energy consumption, and higher power consumption compared with Mylinear through efficient data management. Note that in Figure 5(a), there are larger throughput variations when the gamma parameter is 1. The reason is that when the data locality is high, the OS buffer cache can kick in and make the overall throughput vary wildly, resulting in a bi-modal distribution [Joukov et al. 2006; Tarasov et al. 2011]. We have rerun this experiments ten times more, plotted a histogram, and verified that there were two throughput modes: one from the RAM buffer cache and a second from the low-level tiering hybrid drive. Figure 5(d) shows that the HDD is spun down around 10% of the time when Gamma is small. The reason is that when the Logical Blocks (LBs) are more narrowly distributed, GreenDM has a larger chance to spin down the HDD. Note that, although GreenDM shows the HDD being spun down, it incurs only a small number of HDD start-stop cycles that can be ignored towards the HDD's endurance reduction. Figures 5(e) and 5(f) show that: (1) the HDD-only drive does not reduce the SSD's endurance; (2) the SSD-only drive wears out the SSD to a moderate degree compared with tiering hybrids; (3) GreenDM configurations wear out the SSD faster than Mylinear; and (4) larger Gamma value tends to wear out the SSD faster. Next we discuss the trade-offs for the tiering hybrid drives.

Higher throughput leads to larger energy savings. We can see from Figures 5(a) and 5(b) that when the throughput is higher, the corresponding energy savings are larger under any condition. The main reasons are that: (1) it takes less time to finish the same amount of work when the throughput is higher; and (2) the system-level average power consumption between GreenDM and Mylinear is close (see Figure 5(c)).

There are trade-offs between performance and power consumption in GreenDM. As shown in Figures 5(a) and 5(c), GreenDM achieves higher throughput (50–267%) compared to Mylinear, but consumes 3% more power. The reason is that because the SSD's I/Os are faster than the HDD, the bottleneck is shifted further to the CPU and RAM, making the whole system more active and consuming more power, even though the HDD is spun down in some degree.

There are trade-offs between performance and SSD endurance. As shown in Figures 5(e) and 5(f), GreenDM achieves higher throughput than Mylinear does, but it wears out the SSD faster, from 4–8×, for the same reasons explained in Section 3.3. Moreover, a larger Gamma value can wear out the SSD faster. The reason is that when the Gamma parameter is larger, I/Os are distributed over a wider range of LBs. Hence, there are more promotions and demotions, which eventually increases the SSD read and write counts and reduces the SSD's endurance.

Different tiering GreenDM tunable parameters have different effects on performance, energy, and device endurance. As shown in Figures 5(a), 5(b), 5(d), 5(e), and 5(f), different MCML values under different data locality affect the performance, energy, and endurance in different ways. For example, when the Gamma and MCML values are 1 and 64, respectively, GreenDM improves throughput by 267%, saves energy by 50%, and wears out the SSD's endurance 4× faster. However, when the Gamma and MCML values are 16 and 4, respectively, GreenDM improves throughput by 93%,

saves energy by 71%, and wears out the SSD 7× faster. The reason is that different GreenDM parameters create different benefits and (CPU and I/O) overhead under different data locality: (1) when the Gamma value is small, larger MCML values can promote hot data to the SSD faster; and (2) when the Gamma value is large, smaller MCML values incurs less CPU and I/O overhead. However, different configurations with different Gamma values wear out the SSD to a different degree. Therefore, to meet different requirements, such tunable parameters have to be chosen carefully.

We now discuss the results of the caching GreenDM. As shown in Figure 5, there are similar trade-offs among performance, energy, power, and device endurance, and similar versatility effects as we discussed above for tiering GreenDM. We then compare caching GreenDM against tiering GreenDM in more details. For this Filebench file-server workload, caching GreenDM achieves less throughput (14–71%) than tiering GreenDM does when the system parameters (i.e., MCML, PT, and Gamma) vary, as we can see in Figure 5(a). The file-server workload has lots of I/Os: both reads and writes. There are more reads than writes, but the difference is not that significant, according to Filebench. That means the overhead of the write back is going to play some role in making the throughput lower. Moreover, a medium MCML achieves the best throughput in caching GreenDM under various Gamma values because it achieves the best balance between data movement benefit and overhead.

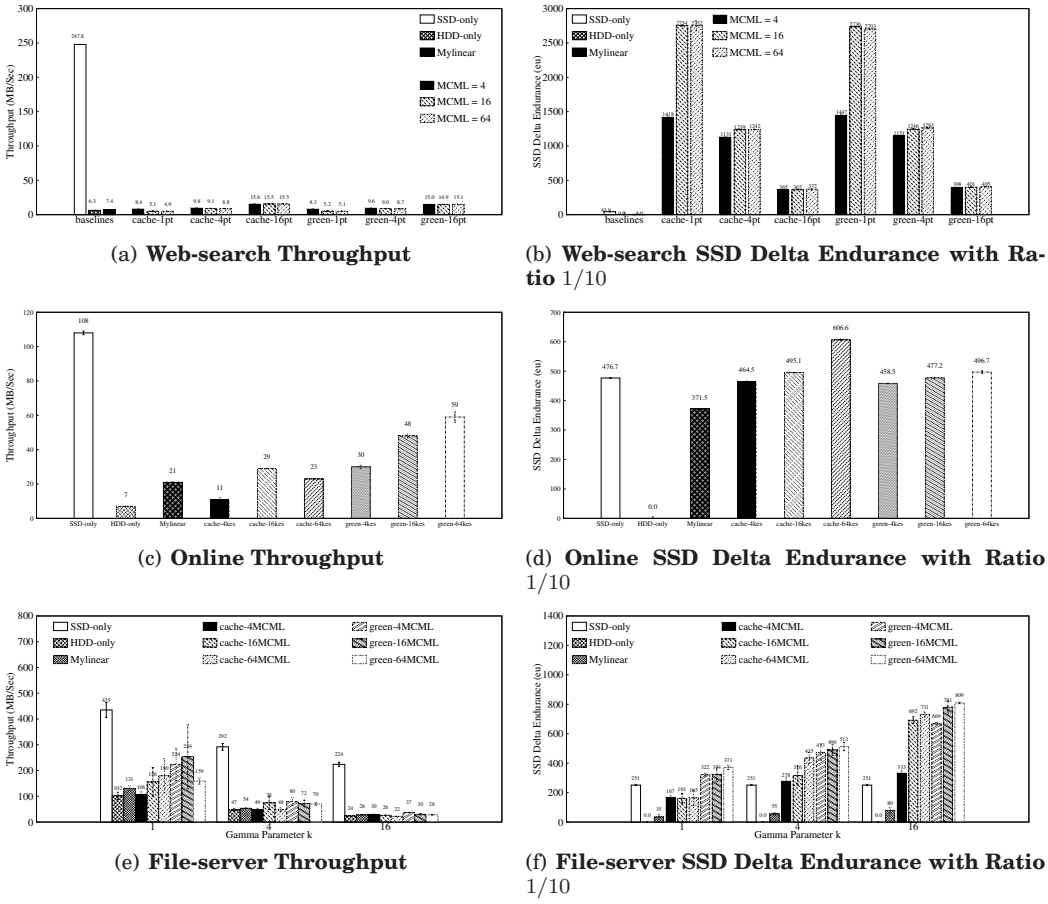
The caching GreenDM also consumes more energy (i.e., up to 57%) than tiering GreenDM does, as shown in Figure 5(b). The reason is similar to that explained in Section 3.4. The caching GreenDM consumes similar amount of power as the tiering GreenDM does, as shown in Figure 5(c). The caching GreenDM spins down the HDD even fewer times than tiering GreenDM does, as shown in Figure 5(d), because the caching GreenDM uses the HDD more aggressively than tiering GreenDM. Lastly, the caching GreenDM wears out the SSD slower (i.e., 27–60% and 28–60%) than the tiering GreenDM does when the ratio of read-to-write is $1/10$ and $1/100$, as seen in Figure 5(e) and Figure 5(f), respectively. The main reason is that the aggregated primary SSD I/Os help reduce the SSD's wear out. Therefore, the caching GreenDM wears out the SSD slower than the tiering GreenDM does. A smaller MCML value wears out the SSD slower in caching GreenDM under various Gamma values since it incurs less SSD accesses.

3.6. Capacity Discussion

The capacity ratio of the SSD as a fraction over the total capacity matters for both the tiering and caching GreenDM in terms of throughput, energy and power, device endurance, and therefore the trade-offs. We therefore configured a new capacity ratio (i.e., $1/8$), and reran all the experiments to show the effects. To avoid duplicated description and discussion, we focus more on representative results (i.e., throughput and SSD endurance reduction). We present the results in Figure 6.

Web-search trace workload. In terms of throughput for the Web-search workload, as we can see from Figures 3(a) and 6(a), when the SSD capacity ratio varies from $1/4$ to $1/8$, the throughputs for both caching and tiering goes down, ranging from 48% to 81%. The main reason for the throughput degradation is due to the increased number of data movements between the SSD and the HDD for both the two systems and a reduced SSD hit ratio when the available SSD capacity is reduced in half. We can also see that when the SSD capacity decreases, the throughput of Mylinear increases by 19% because of fewer SSD hits.

In terms of SSD endurance reduction, as we can see from Figures 3(e) and 6(b), when the SSD capacity ratio varies from $1/4$ to $1/8$, it wears out the SSD faster for both caching and tiering systems, from 26% to 3×. The reason is that the smaller SSD



(a) **Web-search Throughput**

(b) **Web-search SSD Delta Endurance with Ratio** $1/10$

(c) **Online Throughput**

(d) **Online SSD Delta Endurance with Ratio** $1/10$

(e) **File-server Throughput**

(f) **File-server SSD Delta Endurance with Ratio** $1/10$

Fig. 6. **Results of New Capacity Ratio for Three Workloads**.

capacity causes more data movements between the SSD and the HDD for both tiering and caching systems. The smaller the PT value is, the more the data movements are incurred because PT tends to be the dominating factor of SSD's endurance reduction. We can also see that when the SSD capacity decreases, it wears out the SSD 58% slower for Mylinear because of reduced SSD accesses.

Online trace workload. For throughput, as we can see from Figures 4(a) and 6(c), when the SSD capacity ratio varies from $1/4$ to $1/8$, the throughput for the tiering system goes down, from 21% to 39%; throughput for the caching system increases from 7% to 18% when the ES is 4K and 16K, respectively, and decreases by 4% when the ES is 64K. The reason for the tiering system throughput degradation is due to the increased data movements between the SSD and the HDD when the SSD capacity is smaller. The reason for the caching system throughput increase when the ES is small is because: (1) the caching system is bottlenecked by the write-back I/Os since this is a write-intensive workload; and (2) the smaller SSD capacity reduces the number of write-back I/Os. The reason for the small caching system throughput degradation is due to the fact that when the ES is larger, data movements between the SSD and the

HDD causes more overhead. We can also see that when the SSD capacity decreases, the throughput of Mylinear increases by 32% due to the aforementioned reason.

For SSD endurance reduction, as we can see from Figures 4(e) and 6(d), when the SSD capacity ratio varies from $1/4$ to $1/8$, the SSD endurance reduction for both tiering and caching stays roughly the same. The reason is actually due to the net effect of the following factors: (1) the smaller SSD capacity leads to more data movements, which wears out the SSD faster; and (2) the smaller SSD capacity leads to fewer SSD hits, which wears out the SSD slower. We can also see that when the SSD capacity reduces, it wears out the SSD 8% slower for Mylinear due to reduced SSD accesses.

File-server workload. In terms of throughput, as shown in Figures 5(a) and 6(e), when the SSD capacity ratio varies from $1/4$ to $1/8$, the throughputs for both tiering and caching systems generally reduces, from 35% to 57%. The main reason is due to the reduced SSD hits and increased data movements between the SSD and the HDD when the available SSD capacity becomes smaller. We can also see that when the SSD capacity decreases, the throughput of Mylinear increases from 4% to 30% instead of decreasing. We believe that this difference is caused by the degree to which OS's buffer cache is used here.

In terms of SSD endurance reduction, as shown in Figures 5(e) and 6(f), when the SSD capacity ratio varies from $1/4$ to $1/8$, it generally wears out the SSD slower for both tiering and caching, from 7% to 25%. The main reason is because there are fewer SSD hits and it is generally a bigger factor than the increased data movements. We can also see that for Mylinear, when the SSD capacity reduces, it wears out the SSD slower, from 19% to 47%, because of fewer SSD hits since the SSD capacity becomes smaller.

3.7. Evaluation Summary

In this section, we summarize the best configuration for each of workloads in both tiering and caching GreenDM under the current hardware setup. Future storage system designs can potentially refer to the conclusion here.

(1) For the read-intensive Web-search workload, medium Promotion Threshold (PT) and Maximum Concurrent Migration Limit (MCML) values lead to the best throughput for both tiering and caching GreenDM due to the net effect of migration benefit over overhead. In terms of SSD endurance reduction, the PT value is the dominant factor for both tiering and caching GreenDM since it affects the SSD access to a large degree. A smaller PT value wears out the SSD faster since it incurs more SSD accesses.
(2) For the write-intensive Online workload, a larger Extent Size (ES) value leads to higher throughput for tiering GreenDM since it introduces more efficient prefetching. For caching GreenDM, a medium ES value leads to the highest throughput since it best balances the prefetching efficiency and write-back overhead. However, a larger ES value wears out the SSD faster for both tiering and caching GreenDM since it incurs more SSD accesses.
(3) For the read- and write-intensive fileserver workload, the Gamma value is the dominant factor for both tiering and caching GreenDM's throughput and SSD endurance since data locality matters significantly for the two systems to work. For tiering GreenDM, when Gamma is small, a large MCML value achieves better throughput and wears out the SSD slower since it migrates hot I/Os earlier for tiering GreenDM; when Gamma is large, a small MCML value achieves better throughput and wears out the SSD slower since it incurs less overhead for tiering GreenDM. The OS buffer cache also plays an role here. For caching GreenDM, a medium MCML value achieves the best throughput under various Gamma values

since it best balances the data movement benefit and overhead; a smaller MCML value wears out the SSD slower under various Gamma values since it incurs fewer SSD accesses.
(4) For all workloads, since the system (e.g., CPU and RAM) becomes busier due to tiering GreenDM's data management, the system consumes slightly more power than the Mylinear baseline does.
(5) For all workloads, since the system power consumption is not largely different, larger throughput leads to smaller energy consumption when the total amount of workload is fixed.

We now summarize the evaluation results of caching vs. tiering under the current hardware setup.

(1) For the read-intensive Web-search workload, caching achieves similar throughput, energy, and power consumption with the tiering system since the two systems are similar. Moreover, caching wears out the SSD slower than tiering due to the aggregated primary SSD I/Os in the tiering system.
(2) For the write-intensive Online workload, caching achieves less throughput than tiering does due to the negative effect of the write-back policy in the caching system. Moreover, caching causes higher energy and power consumption, and wears out SSD faster than tiering does due to the same negative write-back effect in the caching system.
(3) For the read- and write-intensive fileserver workload, caching has a lower throughput than tiering does due to the negative write-back effect in the caching system. Moreover, caching also wears out the SSD slower because the tiering system has more aggregated primary SSD I/Os. Moreover, caching causes higher energy consumption than tiering, and similar power consumption with tiering.
(4) Caching and tiering are fairly similar except that: (1) caching only maintains mapping information from the cache device to the lower-level device while tiering has to maintain a mapping from the whole virtual device to the actual physical device; (2) caching has to further support a write policy in case of write hit in the cache device while tiering does not need to; and (3) tiering can achieve better initial purchase cost over capacity than caching does since the hot device is used as the primary storage.

Finally, we summarize the results when comparing two SSD capacity ratios (i.e., $1/4$ vs. $1/8$).

(1) For the read-intensive Web-search trace workload, the smaller SSD capacity leads to decreased throughputs and wears out the SSD faster for both tiering and caching GreenDM. This is due to a somewhat decreased SSD usage degree, but primarily due to a larger increase in data movements. It also leads to decreased throughput and wears out the SSD slower for Mylinear due to decreased SSD hits.
(2) For the write-intensive Online trace workload, the smaller SSD capacity causes different throughput effects in tiering and caching GreenDM. For tiering GreenDM, the throughput decreases, while for the caching GreenDM, the throughput generally increases. This is because for this workload, the caching GreenDM is bottlenecked by the write-back I/Os while the tiering GreenDM is bottlenecked by the SSD hit rate and the data movement. The smaller SSD capacity also wears out the SSD at a similar rate for both tiering and caching GreenDM. This is due to the net effect of the reduced SSD hits and the increased data movement. It also leads to decreased throughput and wears out the SSD slower for Mylinear due to reduced SSD hits.

(3) For the read- and write-intensive File-server workload, the smaller SSD capacity generally leads to decreased throughputs and wears out the SSD slower for both tiering and caching GreenDM. This is due to the decreased SSD hit to a large degree and increased data movements to a smaller degree. It also leads to increased throughput for Mylinear due to the OS buffer cache effects and wears out the SSD slower for Mylinear due to reduced SSD accesses.

4. RELATED WORK

Our work is different from past ones in three ways: (1) we compared empirically the tiering and caching modes of the hybrid drive in several dimensions; (2) we developed an endurance model and studied the trade-offs among performance, energy, and device endurance of the hybrid drive in both tiering and caching modes empirically; and (3) we offered a versatile solution to enable the system parameters to be tuned for specific workloads.

There are many existing systems exploring SSD either as a cache [Luo et al. 2012; vFlash 2012; nimble casl 2014; netapp flashpool 2014; bcache 2012; flashcache 2011], where SSDs are used to cache data, or as primary storage [Kim et al. 2011; Guerra et al. 2011; tier 2012; Xie and Sun 2008; Chen et al. 2011; Strunk 2012] since hybrids can achieve better trade-off throughput, capacity, and cost. There are also several hybrid drives in industry: Apple's Fusion Drive [fusion drive 2012], Microsoft's Ready Drive [Panabaker 2006], Western Digital's Solid State Hybrid Drive (SSHD) [wd sshd 2013], Tintri's VMstore [vmstore 2013], Nimble's CASL [nimble casl 2014], and Dell's Compellent Flash Array [dell compellent 2013]. However, they lacks empirical comparison studies for the tiering and caching modes of the hybrid drive. MAID [Colarelli and Grunwald 2002] only briefly discusses the pros and cons of caching and migration based policies for massive storage systems. There is one work [H. Kim and S. Seshadri and C. L. Dickey and L. Chiu 2014] we know that evaluates Phase Change Memories (PCMs) for enterprise storage systems—yet the authors only simulated case studies of caching and tiering approaches. GreenDM is different because it supports both tiering and caching modes with similar strategies and environment to evaluate empirically the pros and cons of the tiering and caching hybrid drives.

Many performance, energy, and endurance relevant studies are based on simulation: HybridStore [Kim et al. 2011], Pearl [Xie and Sun 2008], and NVCache [Bisson et al. 2007]. While simulation can help provide early useful results, we believe that empirical experiments are more realistic. GreenDM performs real-world experiments to study the trade-offs among performance, energy, and endurance of a tiering hybrid drive.

For the SSD endurance metric, past studies normally refer to the number of erasure cycles that can be performed on an SSD during its lifetime [Lee et al. 2012], and do not provide a concrete model and metric to help study the SSD's endurance. One study [V. Mohan and T. Siddiqua and S. Gurumurthi and M. R. Stan 2010] explores a hardware-specific SSD endurance model. While it is useful in some cases, it requires hardware parameters (e.g., voltage, density, etc.) to estimate the endurance through simulation, and can be inconvenient for user-level endurance estimation in reality. GreenDM goes further by developing an endurance model and metric to help study the trade-offs among performance, energy, and endurance in a versatile tiering hybrid drive.

Many storage systems use SSDs as the front tier, but they aim for high performance, efficient energy consumption, or improved endurance. Thus, they often do not closely study the trade-offs among performance, energy, and endurance. Moreover, many studies do not offer flexible policies to enable adaptation to different workloads. HybridStore [Kim et al. 2011] consolidates SSDs and HDDs for a cost-efficient storage system while meeting the performance and lifetime requirements. It is based on simulation, and does not study the trade-offs among performance, energy, and endurance of the

tiering hybrid drive. EDT [Guerra et al. 2011] dynamically migrates a fixed-size extent among different tiers to satisfy performance requirements and reduce power consumption. BTIER [tier 2012] uses a fast storage medium for caching and migrates aged data to a lower tier over time for high performance. Its migration policies are somewhat configurable, but it does not consider the power consumption and the endurance of the tiered storage. Pearl [Xie and Sun 2008] tries to balance the performance, energy, and reliability of disk arrays by migration. It relies on simulations alone and does not empirically study the trade-offs in details. Hystor [Chen et al. 2011] and Aggregate [Strunk 2012] use SSDs as the front tier primary storage for high performance only. NVCache [Bisson et al. 2007] utilizes NVRAM for the I/O subsystem for lower power consumption. GreenHDFS [Kaushik and Bhandarkar 2010] explores how to divide servers in a data-center into different zones to save power while maintaining performance. PDC [Pinheiro and Bianchini 2004] discusses how to migrate data center workloads to fewer disks so that others can be put into lower-power states to save energy. NVCache [Bisson et al. 2007] utilizes NVRAM for the I/O subsystem for lower power consumption. GreenFS [Joukov and Sipek 2008] allows hard disks to be kept off most of time to minimize the disk-drive-related power consumption. MAID [Colarelli and Grunwald 2002] uses data placement, scrubbing, and recovery techniques to put many of the drives in the system into a low-power mode to save energy. Pergamum [Storer et al. 2008] adds NVRAM at each storage node to allow inter-disk data verification while the disk is powered off to save power in a distributed system. PARAID [Weddle et al. 2007] allows adaptive transitions between several different RAID layouts to trade off energy, performance, and reliability. FAWN [Andersen et al. 2009] uses "wimpy" nodes with power-efficient CPUs and I/O capabilities to save power while achieving performance and scalability in a distributed system.

GreenDM is different from the above approaches. It explores in-depth the trade-offs among performance, energy, and endurance in a tiering hybrid drive and comes with a versatile approach so that important system parameters can be investigated and traded off to be best tuned for specific workloads.

5. LIMITATIONS AND FUTURE WORK

While GreenDM estimates the endurance metric by counting the SSD reads and writes and the start-stop (spin-up/down) cycles of the HDD, the endurance metric can be improved. A finer-grained counting in terms of the internal SSD erasure cycles and the FTL's behavior could help build a more accurate endurance estimation for the SSD.

GreenDM provides coarse-grained control (i.e., tunable parameters) to trade-off performance, energy, and device endurance under different workloads. We do not offer QoS guarantees of caps. To reach that goal, we first have to formally study the relationship between performance, energy, device endurance, and various controllable system parameters. We believe machine-learning-based approaches (e.g., hill-climbing [Li et al. 2012a] and control theory [Zhu et al. 2009; Li et al. 2011a]) could help explore such relationships.

GreenDM currently flushes dirty data periodically. It can be dangerous. To provide transaction support, it requires a journaling mechanism for hybrid storage systems.

We currently build the virtual device from two drives only: an SSD and an HDD. We could potentially scale the current setup to multiple drives and more types (e.g., SAS, Shingled, PCM, and NAS) and develop more generalized techniques.

The caching and tiering systems share several design traits. Our current setup better represents a tiering system than a caching one. Caching systems are normally deployed in large storage systems where the caching tier is comparably small compared with the lower-level storage (e.g., 1 PB). We used our current environment because our comparison study makes sense when the hardware and software setup is the same be-

tween the two systems. It would be interesting and challenging to further explore our caching system in a larger storage system.

The trade-off analysis we performed is helpful to further study the cost dimension of our hybrid drive for the gained performance justification [Li et al. 2014].

There are many commercial hybrid products available in industry, as mentioned in Section 4. It would be useful to compare these commercial systems and with ours in the future.

6. CONCLUSION

We designed, built, and evaluated the versatile hybrid drive (both as tiering and caching systems) to study the trade-offs among performance, energy, and endurance. We presented interesting results for various trade-offs observed. In the FIU online trace workload, for example, we showed that tiering GreenDM achieved a higher throughput (58–142%) than Mylinear, but consumed more power (4–8%) and further reduced the SSD's endurance by 11–15% under a certain condition. We observed similar trends for caching GreenDM as well. We also demonstrated the importance of matching tunable parameters to different workloads to better trade-off performance, energy, and endurance. For tiering GreenDM, a larger Extent Size (ES) lead to higher throughput and larger energy savings, but reduced the SSD's endurance further; for caching GreenDM, a too large extent lead to lower throughput and lower energy savings, but it wore out the SSD faster.

Acknowledgements. We thank Vasily Tarasov for help with Filebench. We thank Rajesh Aavuty for help with paper review and experiments preparation. This work was supported in part by NSF awards IIS-1251137 and CNS-1302246.

REFERENCES

D. G. Andersen, J. Franklin, M. Kaminsky, A. Phanishayee, L. Tan, and V. Vasudevan. 2009. FAWN: A Fast Array of Wimpy Nodes. In *Proceedings of the 22nd ACM Symposium on Operating Systems Principles (SOSP '2009)*. ACM SIGOPS, New York, NY, USA, 1–14.

L. A. Barroso and U. Hölzle. 2009. The Datacenter as a Computer: An Introduction to the Design of Warehouse-Scale Machines. *Synthesis Lectures on Computer Architecture* 4, 1 (2009), 1–108.

bcache 2012. Bcache. (2012). http://bcache.evilpiepirate.org/.

T. Bisson, S. A. Brandt, and D. D.E. Long. 2007. A Hybrid Disk-Aware Spin-Down Algorithm with I/O Subsystem Support. In *In Proceedings of the 26th IEEE International Performance, Computing and Communications Conference*. IEEE Computer Society, Los Alamitos, CA, USA, 236–245.

A. Brown and M. Seltzer. 1997. Operating System Benchmarking in the Wake of Lmbench: A Case Study of the Performance of NetBSD on the Intel x86 Architecture. In *Proceedings of the 1997 ACM SIGMETRICS Conference on Measurement and Modeling of Computer Systems*. ACM Press, New York, NY, USA, 214–224.

F. Chen, D. A. Koufaty, and X. Zhang. 2011. Hystor: Making the Best Use of Solid State Drives in High Performance Storage Systems. In *Proceedings of the International Conference on Supercomputing*. ACM, New York, NY, USA, 22–32.

D. Colarelli and D. Grunwald. 2002. Massive Arrays of Idle Disks for Storage Archives. In *Proceedings of the 2002 ACM/IEEE conference on Supercomputing*. ACM, New York, NY, USA, 1–11.

V. Delaluz, A. Sivasubramaniam, M. Kandemir, N. Vijaykrishnan, and M. J. Irwin. 2002. Scheduler-Based DRAM Energy Management. In *Proceedings of the 39th annual Design Automation Conference*. ACM, New York, NY, USA, 697–702.

dell compellent 2013. Dell Compellent Flash Optimized Solutions. (2013). http://www.dell.com/us/business/p/dell-compellent-flash-optimized/pd?~ck=anav.

dm 2012. Device Mapper. (2012). http://en.wikipedia.org/wiki/Device_mapper.

Filebench 2011. Filebench. (2011). http://filebench.sf.net.

FIUTraceRepo 2011. FIU SRCMap Trace Repository. (2011). http://iotta.snia.org/traces/414.

flashcache 2011. Flashcache. (2011). https://github.com/facebook/flashcache/.

fusion drive 2012. Fusion Drive. (2012). http://en.wikipedia.org/wiki/Fusion_Drive.

gamma distribution 2014. Gamma Distribution. (2014). http://en.wikipedia.org/wiki/Gamma_distribution.

J. Gantz and D. Reinsel. 2010. The Digital Universe Decade - Are You Ready? (May 2010). www.emc.com/digital_universe.

B. Guenter, N. Jain, and C. Williams. 2011. Managing Cost, Performance, and Reliability Tradeoffs for Energy-Aware Server Provisioning. In *INFOCOM 2011. 30th IEEE International Conference on Computer Communications, Joint Conference of the IEEE Computer and Communications Societies*. IEEE, Los Alamitos, CA, USA, 1332–1340.

J. Guerra, H. Pucha, J. Glider, W. Belluomini, and R. Rangaswami. 2011. Cost Effective Storage Using Extent Based Dynamic Tiering. In *USENIX FAST*. USENIX Association, Berkeley, CA, 20–34.

S. Gurumurthi, A. Sivasubramaniam, M. Kandemir, and H. Franke. 2003. DRPM: Dynamic Speed Control for Power Management in Server Class Disks. In *Proceedings of the 30th International Symposium on Computer Architecture*. ACM Press, New York, NY, USA, 169–179.

H. Kim and S. Seshadri and C. L. Dickey and L. Chiu. 2014. Evaluating Phase Change Memory for Enterprise Storage Systems: A Study of Caching and Tiering Approaches. In *Proceedings of the 12th USENIX Conference on File and Storage Technologies*. USENIX, Berkeley, CA, 33–45.

W. Jiang, C. Hu, Y. Zhou, and A. Kanevsky. 2008. Are Disks the Dominant Contributor for Storage Failures? A Comprehensive Study of Storage Subsystem Failure Characteristics. In *Proceedings of the Sixth USENIX Conference on File and Storage Technologies (FAST '08)*. USENIX Association, San Jose, CA, 111–125.

N. Joukov and J. Sipek. 2008. GreenFS: Making Enterprise Computers Greener by Protecting Them Better. In *Proceedings of the 3rd ACM SIGOPS/EuroSys European Conference on Computer Systems 2008 (EuroSys 2008)*. ACM, Glasgow, Scotland, 1–14.

N. Joukov, A. Traeger, R. Iyer, C. P. Wright, and E. Zadok. 2006. Operating System Profiling via Latency Analysis. In *Proceedings of the 7th Symposium on Operating Systems Design and Implementation (OSDI 2006)*. ACM SIGOPS, Seattle, WA, 89–102.

Asim Kadav, Mahesh Balakrishnan, Vijayan Prabhakaran, and Dahlia Malkhi. 2009. Differential RAID: Rethinking RAID for SSD Reliability. In *HotStorage '09: Proceedings of the 1st Workshop on Hot Topics in Storage*. ACM, New York, NY, USA, 4:1–4:22.

R. T. Kaushik and M. Bhandarkar. 2010. GreenHDFS: Towards An Energy-Conserving, Storage-Efficient, Hybrid Hadoop Compute Cluster. In *Proceedings of the 2010 International Conference on Power Aware Computing and Systems (HotPower'10)*. USENIX Association, Berkeley, CA, USA, 1–9.

Y. Kim, A. Gupta, B. Urgaonkar, P. Berman, and A. Sivasubramaniam. 2011. HybridStore: A Cost-Efficient, High-Performance Storage System Combining SSDs and HDDs. In *IEEE MASCOTS*. IEEE Computer Society, Washington, DC, USA, 227–236.

J. G. Koomey. 2011. *Growth in Data Center Electricity Use 2005 to 2010*. Technical Report. Standord University. www.koomey.com.

R. Kothiyal, V. Tarasov, P. Sehgal, and E. Zadok. 2009. Energy and Performance Evaluation of Lossless File Data Compression on Server Systems. In *Proceedings of the Second ACM Israeli Experimental Systems Conference (SYSTOR '09)*. ACM, Haifa, Israel, 4:1–4:12.

S. Lee, T. Kim, K. Kim, and J. Kim. 2012. Lifetime Management of Flash-Based SSDs Using Recovery-Aware Dynamic Throttling. In *Proceedings of the Tenth USENIX Conference on File and Storage Technologies (FAST '12)*. USENIX Association, San Jose, CA, 26–26.

S. Li and A. Belay. 2007. cpuidle — Do nothing, efficiently.... In *Proceedings of the Linux Symposium*, Vol. 2. Linux Symposium, Ottawa, Ontario, Canada, 1–10.

Z. Li, A. Desai, C. Bhatt, and E. Zadok. 2012a. vATM: vSphere Adaptive Task Management. In *In Proceedings of the Seventh International Workshop on Feedback Computing (FC'12)*. ACM, New York, NY, USA, 1–7.

Z. Li, K. M. Greenan, A. W. Leung, and E. Zadok. 2012b. Power Consumption in Enterprise-Scale Backup Storage Systems. In *Proceedings of the Tenth USENIX Conference on File and Storage Technologies (FAST '12)*. USENIX Association, San Jose, CA, 6–13.

Z. Li, R. Grosu, K. Muppalla, S. A. Smolka, S. D. Stoller, and E. Zadok. 2011a. Model Discovery for Energy-Aware Computing Systems: An Experimental Evaluation. In *Proceedings of the 1st Workshop on Energy Consumption and Reliability of Storage Systems (ERSS'11)*. IEEE Computer Society, Los Alamitos, CA, USA, 1–6.

Z. Li, R. Grosu, P. Sehgal, S. A. Smolka, S. D. Stoller, and E. Zadok. 2011b. On the Energy Consumption and Performance of Systems Software. In *Proceedings of the 4th Israeli Experimental Systems Conference (ACM SYSTOR '11)*. ACM, Haifa, Israel, 8:1–8:12.

Z. Li, A. Mukker, and E. Zadok. 2014. On the Importance of Evaluating Storage Systems' $Costs. In *Proceedings of the 6th USENIX Conference on Hot Topics in Storage and File Systems (HotStorage'14)*. USENIX, Berkeley, CA, 1–5. to appear.

T. Luo, R. Lee, M. Mesnier, F. Chen, and X. Zhang. 2012. hStorage-DB: Heterogeneity-Aware Data Management to Exploit the Full Capability of Hybrid Storage Systems. *Proceedings of the VLDB Endowment* 5, 10 (2012), 1076–1087.

M. Jung and M. Kandemir. 2013. Revisiting Widely Held SSD Expectations and Rethinking System-Level Implications. In *Proceedings of the ACM SIGMETRICS/International Conference on Measurement and Modeling of Computer Systems (SIGMETRICS '13)*. ACM, New York, NY, USA, 203–216.

M. Wei and L M. Grupp and F. E. Spada and S. Swanson. 2011. Reliably Erasing Data from Flash-Based Solid State Drives. In *Proceedings of the 9th USENIX Conference on File and Stroage Technologies (FAST '11)*. USENIX Association, Berkeley, CA, USA, 8–22.

netapp flashpool 2014. NetApp Flash Pool. (2014). http://www.netapp.com/us/products/platform-os/flashpool.aspx.

nimble casl 2014. Nimble's Hybrid Storage Architecture. (2014). www.nimblestorage.com/products/architecture.php.

P. Desnoyers. 2013. What Systems Researchers Need to Know about NAND Flash. In *HotStorage '13: Proceedings of the 5th USENIX Workshop on Hot Topics in Storage*. USENIX Association, Berkeley, CA, USA, 1–5.

R. Panabaker. 2006. Hybrid Hard Disk and Ready-Drive Technology: Improving Performance and Power for Windows Vista Mobile PCs. (2006). http://www.microsoft.com/whdc/winhec/pres06.mspx.

E. Pinheiro and R. Bianchini. 2004. Energy Conservation Techniques for Disk Array-Based Servers. In *Proceedings of the 18th International Conference on Supercomputing (ICS 2004)*. ACM, New York, NY, USA, 68–78.

E. Pinheiro, W. Weber, and L. A. Barroso. 2007. Failure Trends in a Large Disk Drive Population. In *Proceedings of the Fifth USENIX Conference on File and Storage Technologies (FAST '07)*. USENIX Association, San Jose, CA, 17–28.

R. Freitas. 2009. Storage Class Memory: Technology, Systems and Applications. In *Proceedings of the 2009 ACM SIGMOD International Conference on Management of Data (SIGMOD '09)*. ACM, New York, NY, USA, 985–986.

G. Schulz. 2007. Storage Industry Trends and IT Infrastructure Resource Management (IRM). (2007). www.storageio.com/DownloadItems/CMG/MSP_CMG_May03_2007.pdf.

G. Soundararajan, V. Prabhakaran, M. Balakrishnan, and T. Wobber. 2010. Extending SSD Lifetimes with Disk-Based Write Caches. In *FAST'10: Proceedings of the 8th USENIX Conference on File and Storage Technologies*. USENIX Association, Berkeley, CA, USA, 8–20.

M. W. Storer, K. M. Greenan, E. L. Miller, and K. Voruganti. 2008. Pergamum: Replacing Tape with Energy Efficient, Reliable, Disk-based Archival Storage. In *Proceedings of the Sixth USENIX Conference on File and Storage Technologies (FAST '08)*. USENIX Association, San Jose, CA, 1:1–1:16.

J. D. Strunk. 2012. Hybrid Aggregates: Combining SSDs and HDDs in a Single Storage Pool. *SIGOPS Oper. Syst. Rev.* 46, 3 (2012), 50–56.

V. Tarasov, S. Bhanage, E. Zadok, and M. Seltzer. 2011. Benchmarking File System Benchmarking: It *IS* Rocket Science. In *Proceedings of HotOS XIII:The 13th USENIX Workshop on Hot Topics in Operating Systems*. USENIX Association, Napa, CA, 8–13.

tier 2012. A Tiered Block Device. (2012). http://sourceforge.net/projects/tier/.

UMassTraceRepo 2009. UMass Trace Repository. (2009). http://traces.cs.umass.edu.

V. Mohan and T. Siddiqua and S. Gurumurthi and M. R. Stan. 2010. How I Learned to Stop Worrying and Love Flash Endurance. In *Proceedings of the 2nd USENIX Conference on Hot Topics in Storage and File Systems (HotStorage'10)*. USENIX Association, Berkeley, CA, USA, 8–13.

A. Verma, R. Koller, L. Useche, and R. Rangaswami. 2010. SRCMap: Energy Proportional Storage Using Dynamic Consolidation. In *Proceedings of the 8th USENIX Conference on File and Storage Technologies (FAST'10)*. USENIX Association, Berkeley, CA, USA, 8–20.

vFlash 2012. Virtual Flash Tech Preview. (2012). http://blogs.vmware.com/vsphere/2012/12/virtual-flash-vflash-tech-preview.html.

vmstore 2013. Tintri VMStore. (2013). www.tintri.com/resources/videos/introduction-to-tintri/.

Wattsup 2010. Watts up? PRO ES Power Meter. (2010). www.wattsupmeters.com/secure/products.php.

wd sshd 2013. WD Blue. (2013). http://www.wd.com/en/products/products.aspx?id=800#tab11.

C. Weddle, M. Oldham, J. Qian, A. A. Wang, P. Reiher, and G. Kuenning. 2007. PARAID: a Gear-Shifting Power-Aware RAID. In *Proceedings of the Fifth USENIX Conference on File and Storage Technologies (FAST '07)*. USENIX Association, San Jose, CA, 245–260.

A. W. Wilson. 2008. Operation and implementation of random variables in Filebench. (2008).

C. P. Wright, N. Joukov, D. Kulkarni, Y. Miretskiy, and E. Zadok. 2005. Auto-pilot: A Platform for System Software Benchmarking. In *Proceedings of the Annual USENIX Technical Conference, FREENIX Track*. USENIX Association, Anaheim, CA, 175–187.

T. Xie and Y. Sun. 2008. PEARL: Performance, Energy, and Reliability Balanced Dynamic Data Redistribution for Next Generation Disk Arrays. In *IEEE MASCOTS*. IEEE Computer Society, Washington, DC, USA, 8–16.

zcav 2014. ZCAV. (2014). http://en.wikipedia.org/wiki/Zone_bit_recording.

Q. Zhu, Z. Chen, L. Tan, Y. Zhou, K. Keeton, and J. Wilkes. 2005. Hibernator: Helping Disk Arrays Sleep Through the Winter. In *Proceedings of the 20th ACM Symposium on Operating Systems Principles (SOSP '05)*. ACM Press, Brighton, UK, 177–190.

X. Zhu, M. Uysal, Z. Wang, S. Singhal, A. Merchant, P. Padala, and K. Shin. 2009. What does control theory bring to systems research? *SIGOPS Oper. Syst. Rev.* 43, 1 (2009), 62–69.

Y. Zhu and F. Mueller. 2004. Feedback EDF Scheduling Exploiting Dynamic Voltage Scaling. In *RTAS '04: Proceedings of the 10th IEEE Real-Time and Embedded Technology and Applications Symposium*. IEEE Computer Society, Washington, DC, USA, 33 – 63.

Received May 2014; revised December 2014; accepted December 2014