RZ 3769 (# 99779) 03/26/2010
Computer Science 20 pages

Research Report

Performance of the Greedy Garbage-Collection Scheme in Flash-Based Solid-State Drives

Ilias Iliadis

IBM Research – Zurich
8803 Rüschlikon
Switzerland

Email: ili@zurich.ibm.com

Research
Almaden · Austin · Beijing · Delhi · Haifa · T.J. Watson · Tokyo · Zurich

Performance of the Greedy Garbage-Collection Scheme in Flash-Based Solid-State Drives

Ilias Iliadis

IBM Research – Zurich, 8803 Rüschlikon, Switzerland
Phone: +41-1-724-8646; Fax: +41-1-724-8952; e-mail: ili@zurich.ibm.com

Abstract

In flash-based solid-state drives (SSD) and log-structured file systems, new data is written out-of-place, which over time exhausts the available free space. New free space is created by the garbage-collection process, which reclaims the space occupied by invalidated data. The write amplification, incurred because of the additional write operations performed by the garbage-collection mechanism is a critical factor that negatively affects the lifetime and endurance of SSDs. A theoretical model is developed to evaluate the impact of the greedy garbage-collection mechanism on the performance of large storage systems. The system operation and behavior are comprehensively characterized for uniformly-distributed random small user writes. Results of theoretical and practical importance are analytically derived and confirmed by means of simulation. Closed-form expressions are derived for both the number of relocated pages and the write amplification. The write amplification is analytically assessed for the key system parameters, i.e., the total system memory space, the proportion of the memory space occupied by valid user data, and the block size in terms of number of pages. Our results demonstrate that as the system occupancy increases, the write amplification increases. Furthermore, as the number of pages contained in a block increases, the write amplification increases and approaches an upper bound. They also show that the number of free pages reclaimed by the greedy garbage-collection mechanism after each block recycling takes one of two successive values, which provides a quasi-deterministic performance guarantee.

1. Introduction

A current trend in the data storage industry is the increasing adoption of non-volatile NAND-flash memories, such as solid-state drives (SSD), that provide random I/O performance and access latency that are orders of magnitude better than those of rotating hard-disk drives [1, 2]. Also, owing to their good characteristics in terms of power consumption and shock resistance, these memories have been widely deployed in portable devices.

Data is read and written in a unit of one page, and erased in a unit of one block, which contains several pages. Typically, a block contains 64 or 128 pages, with the page size equal to 2 or 4 KB. SSDs require out-of-place writes, in that new data does not overwrite the memory location where the data is currently stored. Write-in-place is not used because it would require reading, erasing, and rewriting of the entire corresponding block, which would lead to performance and endurance degradation. As updated data is written in available free pages, the pages storing old data are invalidated. As a result, the available free space is gradually exhausted, which therefore over time renders the creation of new free pages necessary for subsequent write operations. This task is performed by the garbage-collection process, a process that is also required in log-structured file systems, disks, and arrays [3, 4].

The garbage-collection mechanism first identifies blocks for cleaning based on a given policy. Then valid data residing in these blocks is copied (relocated) to other blocks, and finally the blocks are erased so that they become available for rewriting. Consequently, this mechanism introduces additional read and write operations, the extent of which depends on the specific policy deployed, as well as on the system parameters. These additional writes result in the multiplication of user writes, a phenomenon referred to as “write amplification”. As the number of erase/write operations that can be performed before an SSD wears out is limited, the extent of the write amplification is critical because it negatively affects the lifetime and endurance of SSDs. Therefore, a garbage-collection mechanism is efficient when it keeps the write amplification as low as possible, and also achieves a good wear leveling in the sense of blocks being worn out as evenly as possible. To achieve these goals, various garbage-collection policies have been proposed in the literature, such as the “greedy” and “cost-benefit” policies [3, 4]. In this report we consider the greedy policy, which selects blocks with the smallest number of valid pages, such that it yields the most amount of free space for reclaiming. Numerous simulation runs support the claim that, under a random small write workload, this policy minimizes the write amplification. To date, however, this remains a conjecture as it has not yet been theoretically proved.

The key contributions of this work are the following. We develop a theoretical model to capture the operational characteristics of the greedy garbage-collection scheme and assess its impact on the performance of large storage systems. The usefulness of this model follows from the fact that, owing to the state and memory explosion, neither Markov chain models nor simulators can effectively be used to assess the performance of large flash-memory storage systems. We analytically derive closed-form expressions that relate the write amplification, along with other performance measures of interest, to the key system parameters, i.e., the total system memory space, the proportion of the memory space occupied by valid user data, and the block size in terms of number of pages. Our results demonstrate that as the system occupancy, which is the ratio of the number of pages containing valid user data to the total number of pages available in the system, increases, the write amplification increases. Furthermore, as the number of pages contained in a block increases, the write amplification increases and approaches an upper bound. The results obtained also reveal that the number of free pages reclaimed by the greedy garbage-collection mechanism after each block recycling takes one of two specific successive values that depend on the system parameters. This result is of practical importance in time-critical systems because it implies that a block erase triggered by the garbage collection process may block a real-time task for an essentially fixed period of time, which provides a quasi-deterministic performance guarantee. The analytical findings of this work are also supported by simulation results, which shed additional light on the behavior and effectiveness of the greedy garbage-collection policy. They demonstrate that the greedy policy can be implemented in such a way that blocks are evenly recycled, and therefore inherently provides a good degree of wear leveling.

The remainder of the report is organized as follows. Section 2 provides a survey of the relevant literature on the greedy garbage-collection scheme. Section 3 presents the relevant system parameters, describes the operation of the garbage-collection process, and demonstrates that the write amplification depends only on the average number of relocated pages, which in turn depends on the system occupancy. In Section 4, an analytical performance model of the greedy garbage-collection policy is developed, and a closed-form expression for the distribution of the number of relocated pages is derived. Section 5 presents numerical results demonstrating the impact of the system parameters on the write amplification. The analytical findings are also supported by simulation results, which provide further insight into the behavior of the greedy garbage-collection policy. Finally, we conclude in Section 6.

2. Related Work

Approximate analytical methods for deriving the write amplification corresponding to the greedy garbage-collection policy have been presented in [5, 6, 7] assuming uniformly-distributed random small user writes. For a system comprised of a large number of blocks and a large number of pages per block, it was shown in [5] that the write amplification depends only on the system occupancy or utilization, which is the ratio of the number of pages containing valid user data to the total number of pages available in the system. More specifically, an analytic expression for the system occupancy was derived as a function of the write amplification, based on the assumption that the block selected by the greedy policy always contains the same number of valid pages. Here we extend this result by analytically deriving the write amplification as a function of both the system occupancy and the number of pages per block. Furthermore, we show that the blocks selected by the greedy policy do not always contain the same number of valid pages, but that this number takes one of two specific successive values that depend on the system parameters.

An approximate analytical method was developed in [6] for assessing the write amplification for a windowed greedy policy, which includes the greedy policy as a special case in which the window covers all blocks of the system. This method allows the numerical evaluation of the write amplification for systems up to a certain size that depends on the amount of computational resources available. It turns out that this method provides optimistic results, and that for large values of the system occupancy, it significantly underestimates the write amplification. An exact Markov chain model for the assessment of the write amplification was presented in [7]. This model is useful to numerically obtain the write amplification in small-sized systems. It cannot, however, be applied in the case of large systems because of the state explosion of the underlying Markov chain. By contrast, the analytical method presented here assesses the write amplification in large storage systems. Furthermore, it offers the possibility of gaining additional performance insight regarding several aspects of the system operation. We also present simulation results that confirm the analytical results obtained over the entire range of system occupancies. The simulation method, however, has its own limitations; it, too, cannot be applied in the case of very large systems because of the explosion of the corresponding memory required. The fact that neither Markov chain models nor simulators can be used to assess the performance of large flash-memory storage systems, establishes the usefulness of the theoretical model developed and presented by this work.

In flash-memory storage systems with real-time hard constraints, it is essential to provide a deterministic performance guarantee [8]. As a block erase triggered by the garbage-collection process may block a real-time task, it is therefore desirable in time-critical systems to characterize the behavior of the garbage-collection process employed. In [8] it is argued that it is crucial to predict the number of free pages reclaimed after each block recycling so that the system will never be blocked for an unpredictable length of time because of garbage collection. Our results also address this issue in that they demonstrate that, for uniformly-distributed random writes, the greedy garbage-collection policy results in a quasi-deterministic number of free pages reclaimed after each block recycling.

3. System Analysis

The notation used for the purpose of our analysis is given in Table 1. The parameters are divided into two sets, namely, the set of independent and that of dependent parameters, listed in the upper and lower part of the table, respectively.

Each block contains c pages and the system is assumed to contain a total of b blocks, denoted by $b_1, b_2, \ldots, b_b$. The number of pages N containing valid user data is given by

$$N \; = \; u\,c\,, \tag{1}$$

where u is the user storage capacity expressed in number of blocks. A proper system operation requires that $u \leq b - n_{\mathrm{cl}}$, where n_{cl} is the number of blocks kept in a clean state reserved for use by the garbage-collection process.

Let $v_i(t)$ denote the number of valid pages contained in block b_i at time t. Clearly, the sum of all valid pages over all blocks is always equal to the number of pages containing valid user data, that is,

$$\sum_{i=1}^{b} v_i(t) \; = \; N\,, \quad \forall\, t \in [0, \infty)\,. \tag{2}$$

Table 1: Notation of system parameters

Parameter	**Definition**
b	Total storage capacity in number of blocks
u	User storage capacity in number of blocks
c	Number of pages per block
n_{cl}	Number of clean blocks
b_i	ith block $(1 \leq i \leq b)$
$v_i(t)$	Number of valid pages contained in block b_i at time t, $(1 \leq i \leq b)$
N	Number of pages containing valid user data
ρ	System occupancy (utilization)
$\rho_i(t)$	Occupancy (utilization) of block b_i at time t
A	Write amplification
$\{V_i\}$	Sequence of number of relocated pages, $i = 1, 2, \ldots$
V	Number of relocated pages in steady state
v^*	Normalized average number of relocated pages, or mean occupancy (utilization) of garbage-collected blocks

The occupancy or utilization $\rho_i(t)$ of block b_i at time t is given by

$$\rho_i(t) \triangleq \frac{v_i(t)}{c} , \quad \forall\, t \in [0, \infty) , \tag{3}$$

the ratio of valid pages at time t to the total number of pages of block b_i. From (1), (2), and (3), it follows that

$$\sum_{i=1}^{b} \rho_i(t) = \sum_{i=1}^{b} \frac{v_i(t)}{c} = \frac{1}{c} \sum_{i=1}^{b} v_i(t) = \frac{N}{c} = u , \quad \forall\, t \in [0, \infty) . \tag{4}$$

The system occupancy or utilization ρ expresses the ratio of the number of pages containing valid user data to the total number of pages available and is given by

$$\rho \triangleq \frac{u}{b} , \tag{5}$$

the ratio of user storage capacity to total storage capacity. From (3) and (5), it follows that

$$\frac{\sum_{i=1}^{b} \rho_i(t)}{b} = \frac{u}{b} = \rho , \quad \forall\, t \in [0, \infty) , \tag{6}$$

that is, at any time, the average block occupancy is, as expected, equal to the system occupancy. The overprovisioning factor O_f, which is the ratio of b to u, is given by

$$O_f \triangleq \frac{b}{u} = \frac{1}{\rho} . \tag{7}$$

3.1. The Garbage-Collection Process

The garbage-collection process requires that there is always a clean block for page relocation, that is, $n_{\mathrm{cl}} \geq 1$. It turns out that for large values of b, the performance measures are practically insensitive to n_{cl} [4]. For the purpose of our analysis, it therefore suffices to consider $n_{\mathrm{cl}} = 1$, which implies that there is at most one clean block, and that the page-writing and the garbage-collection process are operating in a coordinated fashion. More specifically, every time, say t_i, a block is fully written, the garbage-collection process selects a block for relocation denoted by b_{r_i}, and copies its $V_i = v_{r_i}(t_i)$ valid pages to the clean block denoted by b_{c_i}. The greedy policy reclaims the most amount of free space by selecting a block with the smallest number of valid pages, that is,

$$V_i \;=\; \min_{\substack{1 \leq j \leq b \\ j \neq c_i}} \; v_j(t_i) \; . \tag{8}$$

As, in general, there are multiple blocks containing V_i valid pages, blocks should be selected in such a way that the wear across all blocks is even. One of various ways to achieve a good degree of wear leveling is the following. Blocks are maintained in a queue according to the order in which they are written. An index is assigned to each position in the queue with the clean block having index one, the oldest block having index two and the youngest block having index b. Thus, this queue maintains the relative, not absolute, age of the blocks, with the occupied blocks ordered accordingly in positions $2, 3, \ldots, b$. In particular, the greedy garbage-collection policy selects the oldest of the blocks that have V_i valid pages. Consequently, b_{r_i} is the block with the smallest block-age index that has V_i valid pages. When the relocation of the V_i valid pages has completed, b_{r_i} is recycled as a clean block after being erased, whereas b_{c_i} is no longer clean. Thus, the number of clean blocks remains one. Also note that in the subsequent time interval $(t_i, t_{i+1}]$ the remaining $c - V_i$ free pages of block b_{c_i} are written. From the preceeding, it follows that $b_{r_1}, \ldots, b_{r_i}, \ldots$ denote the sequence of successive blocks selected at times $t_1, \ldots, t_i, \ldots$ by the garbage-collection process for recycling, and that $V_1, \ldots, V_i, \ldots$ denote the corresponding numbers of valid pages that are relocated on the respective clean blocks denoted by $b_{c_1}, \ldots, b_{c_i}, \ldots$.

3.2. Write Amplification

The write amplification A is defined as the average of the actual number of (system) page writes per user page write, and is therefore given by

$$A \;=\; \lim_{i \to \infty} \frac{\displaystyle\sum_{k=1}^{i} c}{\displaystyle\sum_{k=1}^{i} (c - V_k)} \;=\; \lim_{i \to \infty} \frac{c}{c - \frac{\sum_{k=1}^{i} V_k}{i}} \;=\; \frac{c}{c - E(V)} \;=\; \frac{c}{c - \bar{V}} \; , \tag{9}$$

where $\bar{V}$ denotes the average number of relocated pages. From (9), it follows that

$$A = \frac{1}{1 - v^*} , \tag{10}$$

where v^* denotes the normalized average number of relocated pages, which is equal to the mean occupancy of the blocks selected by the garbage-collection process for relocation, given by

$$v^* \triangleq \frac{\bar{V}}{c} , \quad \text{with} \quad 0 \leq v^* \leq 1 . \tag{11}$$

Thus, to assess the write amplification it suffices to derive the average number of relocated pages.

4. Analysis of Large Systems

We proceed to derive the average number of relocated pages $\bar{V}$ as a function of the system parameters for a random write workload, that is, for uniformly-distributed random small user writes. From the definitions given in Table 1, it follows that the probability that a small user write results in an update of a given page is equal to $1/N$. Let us now define by $K_j(t)$ the number of blocks containing j valid pages at time t. Then it holds that

$$\sum_{j=0}^{c} K_j(t) = b , \quad \forall\, t \in [0, \infty) , \tag{12}$$

and

$$\sum_{j=0}^{c} j\, K_j(t) = N = u\, c , \quad \forall\, t \in [0, \infty) . \tag{13}$$

Note also that the probability $p_j(t)$ that a randomly selected block at time t contains j valid pages is given by

$$p_j(t) = \frac{K_j(t)}{b} , \quad \text{for} \quad j = 0, 1, \ldots, c . \tag{14}$$

Let us now consider a typical interval $(t_i, t_{i+1}]$ in which $c - V_i$ user write operations (page updates) are performed in block $b_{c_i}(t_i)$. Note that at the end of this interval, the probability that b_{c_i} contains c valid pages is given by

$$\begin{aligned} P(v_{c_i}(t_{i+1}) = c) = & \left(1 - \frac{V_i}{N}\right)\left(1 - \frac{V_i + 1}{N}\right) \cdots \left(1 - \frac{c-1}{N}\right) \\ & \geq \left(1 - \frac{c}{N}\right)^{c - V_i} \geq \left(1 - \frac{1}{u}\right)^{c} . \end{aligned} \tag{15}$$

Consequently,

$$\lim_{\frac{u}{c} \to \infty} P(v_{c_i}(t_i) = c) \geq \lim_{\frac{u}{c} \to \infty} \left(1 - \frac{1}{u}\right)^{c} = 1 , \quad \text{for} \quad i = 1, 2, \ldots, . \tag{16}$$

Therefore, for any given occupancy ρ,

$$\lim_{\frac{u}{c}\to\infty} P(v_{c_i}(t_i) = c) \ = \ 1 \ , \quad \text{for} \ \ i = 1, 2, \ldots, \ , \tag{17}$$

which implies that as N (or b) increases, while keeping c fixed, the pages of the newly written blocks are all valid. Clearly, on the one hand, owing to the uniform distribution of the random page write requests, the probability that a user write operation invalidates a given page is equal to $1/N$, which for large values of N tends to zero. But, on the other hand, a user write operation invalidates a page of a block. However, the probability that a subsequent user write operation in the interval considered also invalidates a page belonging to the same block tends to zero. Therefore, the probability that two or more of the $c - V_i$ user write operations in the interval $(t_i, t_{i+1}]$ invalidate pages belonging to the same block tends to zero. This implies that for any block, the number of valid pages may change within this interval by at most one. As the garbage-collection process always selects a block with the smallest number of valid pages, this process practically eliminates the possibility of having blocks with a small number of valid pages. Consequently, there exists a number c^*, referred to as *critical number of pages*, such that there are practically no blocks containing c^* or fewer valid pages, that is

$$\lim_{\substack{t\to\infty \\ \frac{u}{c}\to\infty}} p_j(t) = 0 \ , \quad \text{for} \ \ j = 0, 1, \ldots, c^* \ , \tag{18}$$

and

$$\lim_{\substack{t\to\infty \\ \frac{u}{c}\to\infty}} p_j(t) > 0 \ , \quad \text{for} \ \ j = c^* + 1, \ldots, c \ . \tag{19}$$

Note also that at any time t, and owing to the uniform distribution of the random page write requests, the probability $h_j(t)$ that a newly written page invalidates a page located in a block containing j valid pages, referred to as j-block, is given by

$$h_j(t) \ = \ \frac{j \ K_j(t)}{N} \ , \quad \text{for} \ \ j = 0, 1, \ldots, c \ , \quad \forall \, t \in [0, \infty) \ . \tag{20}$$

Substituting (1) into (20), and using (5) and (14) yields

$$h_j(t) \ = \ \frac{j \ p_j(t)}{\rho \ c} \ , \quad \text{for} \ \ j = 0, 1, \ldots, c \ , \quad \forall \, t \in [0, \infty) \ . \tag{21}$$

In particular, for $j = c^* + 1$, we get

$$h_{c^*+1}(t) \ = \ \frac{(c^* + 1) \ p_{c^*+1}(t)}{\rho \ c} \ , \quad \forall \, t \in [0, \infty) \ . \tag{22}$$

From (19) and (22), it follows that $h_{c^*+1}(t) > 0$, $\forall \, t \in [0, \infty)$, which implies that there will always be pages invalidated in blocks containing $c^* + 1$ valid pages, resulting in blocks containing c^* valid pages after the invalidation. Thus, as time progresses, blocks that contain c^* valid pages will always appear. Their number, however, will be negligible compared with the total number of blocks, such that the probability $p_{c^*}(t)$ of randomly selecting one of them at time t will be, according to (18), equal to zero. The garbage-collection process, by contrast, will always select these blocks for relocation because they have the smallest number of valid pages. Moreover, at times when there are no such blocks, there will always be a block with $c^* + 1$ valid pages selected for relocation, because, according to (19), such blocks always exist. From the preceding, it follows that in steady state, that is, for large values of t, a block selected for relocation will always contain either c^* or $c^* + 1$ valid pages. Let V represent the number of relocated pages in steady state, that is,

$$V \triangleq \lim_{i \to \infty} V_i \, . \tag{23}$$

Also, let q $(0 < q \leq 1)$ denote the probability that the number of relocated pages is equal to c^*. Then the distribution of V is a discrete bimodal distribution, that is,

$$\lim_{i \to \infty} P(V_i = j) \; = \; P(V = j) \; = \; \begin{cases} q \, , & j = c^* \\ 1 - q \, , & j = c^* + 1 \\ 0 \, , & \text{otherwise} \, . \end{cases} \tag{24}$$

Note that $q > 0$ because, as discussed above, blocks that contain c^* valid pages will always appear. Also, q can be equal to one in that for large i's, at all $\{t_i\}$ times there will be at least one block with c^* valid pages. The number of such blocks, however, will be negligible compared with the total number of blocks, such that $p_{c^*}(t_i) = 0$ according to (18). From (24), it follows that the average number $\bar{V}$ of relocated pages is given by

$$\bar{V} \; = \; E(V) \; = \; c^* + 1 - q \, . \tag{25}$$

We now proceed to obtain the expected number $E(K_j(t_{i+1}))$ of blocks containing j $(j > c^*)$ valid pages at time t_{i+1} conditioning on the number $K_j(t_i)$ of blocks containing j valid pages at time t_i and on the number V_i of relocated pages. First, we note that the number of j-blocks may change by at most $c - V_i$ in the interval $(t_i, t_{i+1}]$, that is

$$|K_j(t) - K_j(t_i)| \; \leq \; c - V_i \, , \quad \forall \, t \in [t_i, t_{i+1}] \, , \tag{26}$$

with the equality holding if and only if either all page invalidations occur in j-blocks, resulting in $K_j(t_{i+1}) = K_j(t_i) - (c - V_i)$, or all page invalidations occur in $(j+1)$-blocks, resulting in $K_j(t_{i+1}) = K_j(t_i) + (c - V_i)$. From (1), (20), and (26), it follows that

$$|h_j(t) - h_j(t_i)| \; \leq \; \frac{j(c - V_i)}{c \, u} \; \leq \; \frac{c - V_i}{u} \, , \quad \text{for} \;\; j = c^*{+}1, \ldots, c \, , \quad \forall \, t \in [t_i, t_{i+1}] \, . \tag{27}$$

Thus, for large values of u/c it holds that

$$|h_j(t) - h_j(t_i)| \;\le\; \lim_{\frac{u}{c}\to\infty} \frac{c - V_i}{u} \;=\; 0\;, \quad \text{for } \; j = c^* + 1, \ldots, c\;, \quad \forall\, t \in [t_i, t_{i+1}]\;, \tag{28}$$

or

$$h_j(t) = h_j(t_i)\;, \quad \text{for } \; j = c^* + 1, \ldots, c\;, \quad \forall\, t \in [t_i, t_{i+1}]\;. \tag{29}$$

As in the interval $[t_i, t_{i+1}]$ each new page write invalidates a page of a j-block with probability $h_j(t)$, the expected number of j-blocks that will be affected is equal to $(c - V_i)\, h_j(t)$. Thus, the number of j-blocks is expected to be reduced by $(c - V_i)\, h_j(t)$, which will result in an expected increase of the number of $(j - 1)$-blocks. Similarly, the number of j-blocks is expected to increase by $(c - V_i)\, h_{j+1}(t)$, owing to the expected reduction of the $(j + 1)$-blocks. Also, the selection of block b_{r_i}, which contains V_i valid pages for relocation, by the garbage-collection process will reduce the number of V_i-blocks by one. Furthermore, the relocation of the V_i valid pages in the clean block b_{c_i}, and given that, according to (17), b_{c_i} contains c valid pages at time t_{i+1}, results in an increase of the number of c-blocks by one. Consequently,

$$E(K_j(t_{i+1})|V_i) = \begin{cases} K_j(t_i) - (c - V_i)\, h_j(t_i) + (c - V_i)\, h_{j+1}(t_i)\;, & V_i < j < c \\ K_j(t_i) - (c - V_i)\, h_j(t_i) + (c - V_i)\, h_{j+1}(t_i) - 1\;, & j = V_i \\ K_j(t_i) - (c - V_i)\, h_j(t_i) + 1\;, & j = c\;. \end{cases} \tag{30}$$

Unconditioning on V_i, after some manipulations, (24) yields

$$\begin{aligned} &E(K_j(t_{i+1})|K_j(t_i)) \\ &\quad= \sum_{n=0}^{c-1} E(K_j(t_{i+1})|V_i = n)\; P(V_i = n) \\ &\quad= \sum_{n=0}^{j-1} E(K_j(t_{i+1})|V_i = n)\; P(V_i = n) + E(K_j(t_{i+1})|V_i = j)\; P(V_i = j) \\ &\qquad+ \sum_{n=j+1}^{c-1} E(K_j(t_{i+1})|V_i = n)\; P(V_i = n) \\ &\quad= \begin{cases} K_j(t_i) - [c - E(V)]\,[h_j(t_i) - h_{j+1}(t_i)] - P(V_i = j)\;, & c^* < j < c \\ K_j(t_i) - [c - E(V)]\, h_j(t_i) + 1\;, & j = c\;. \end{cases} \end{aligned} \tag{31}$$

Unconditioning on $K_j(t_i)$, (31) yields

$$\begin{aligned} &E(K_j(t_{i+1})) \\ &\quad= \begin{cases} E(K_j(t_i)) - [c - E(V)]\,[h_j(t_i) - h_{j+1}(t_i)] - P(V_i = j)\;, & c^* < j < c \\ E(K_j(t_i)) - [c - E(V)]\, h_j(t_i) + 1\;, & j = c\;. \end{cases} \end{aligned} \tag{32}$$

Considering the system in steady state, that is $i \to \infty$, and in turn $t \to \infty$, it holds that $E(K_j(t_{i+1})) = E(K_j(t_i)) = E(K_j)$. By suppressing t_i and making use of (18), (21), and (24), and given that $E(V) < c$, (32) yields

$$\begin{cases} h_j = h_{j+1} \ , & c^* + 1 < j < c \\ h_j = h_{j+1} + \dfrac{1-q}{c - E(V)} \ , & j = c^* + 1 \\ [c - E(V)]\, h_j = 1 \ , & j = c \\ h_j = 0 \ , & 0 \le j \le c^* \ . \end{cases} \tag{33}$$

Solving (33) recursively for h_j yields, by making use of (1) and (20)

$$\lim_{\frac{u}{c} \to \infty} \frac{K_j}{u} = \begin{cases} \dfrac{c}{j\,(c - \bar{V})} \ , & c^* + 1 < j \le c \\ \dfrac{q\,c}{j\,(c - \bar{V})} \ , & j = c^* + 1 \\ 0 \ , & 0 \le j \le c^* \ . \end{cases} \tag{34}$$

Also, from (5), (11), (14), and (34), it follows that in steady state the probability p_j that a randomly selected block contains j valid pages is given by

$$\lim_{\frac{u}{c} \to \infty} p_j = \begin{cases} \dfrac{\rho}{j\,(1 - v^*)} \ , & c^* + 1 < j \le c \\ \dfrac{q\,\rho}{j\,(1 - v^*)} \ , & j = c^* + 1 \\ 0 \ , & 0 \le j \le c^* \ . \end{cases} \tag{35}$$

Substituting (34) into (12), and using (5) yields

$$\sum_{j=c^*+1}^{c} \frac{K_j}{u} = \frac{q\,c}{(c^*+1)\,(c - \bar{V})} + \sum_{j=c^*+2}^{c} \frac{c}{j\,(c - \bar{V})} \ = \ \frac{1}{\rho} \ , \tag{36}$$

or

$$\rho \ = \ \frac{c - \bar{V}}{c \left[\dfrac{q}{c^* + 1} + S(c^* + 2, c) \right]} \ , \tag{37}$$

where

$$S(n, c) \ \triangleq \ \sum_{j=n}^{c} \frac{1}{j} \ . \tag{38}$$

The probability q is given as a function of c^* and the system parameters by the following proposition.

Proposition 1. *It holds that*

$$q \ = \ \frac{(c^*+1)\,[c - (c^*+1) - c\,\rho\,S(c^*+2, c)]}{c\,\rho - (c^*+1)} \ , \tag{39}$$

where $S(n,c)$ is given by (38).

Proof: Immediate by substituting (25) into (37) and solving for q. □

For very low values of the occupancy ρ, and in particular for $\rho \in (0, (1-2/b)\, c]$, such that $N \leq b-2$, there is always at least one block with zero valid pages, and therefore $V_i = 0$ for $i = 1, 2, \ldots$, and $\bar{V} = 0$. In fact, according to (25), $\bar{V} = 0$ if and only if $c^* = 0$ and $q = 1$. Substituting these values into (37), and using (38), we deduce that

$$\bar{V} \;=\; 0 \,, \quad \text{and therefore } \; c^* \;=\; 0 \,, \quad \text{for} \;\; \rho \in [0, \rho_0] \,, \tag{40}$$

where

$$\rho_0 \;=\; \frac{1}{S(1,c)} \,. \tag{41}$$

In particular, for $\rho \in ((1-2/b)\, c, \rho_0]$, there are instances $i_1, i_2, \ldots$ for which $V_i > 0$ for $i = i_1, i_2, \ldots$. Nevertheless, these instances are rare, and as a result $\bar{V} = 0$. Further increasing the occupancy ρ causes c^* to increase in steps. Let $\rho_1, \ldots, \rho_m, \ldots$ denote the various occupancy points at which c^* increases. More specifically, for $\rho \to \rho_m^-$ it holds that $c^* = m-1$, $q \to 0$, and by virtue of (35), $p_m \to 0$, whereas for $\rho = \rho_m$ it holds that $c^* = m$, $q = 1$, and $p_m = 0$. Consequently, the values $\{\rho_m\}$ are obtained from (37) by considering $c^* = m$, $q = 1$, and, according to (25), $\bar{V} = m$, as follows:

$$\rho_m \;=\; \frac{c-m}{c\, S(m+1, c)} \,, \quad \text{for} \;\; m = 1, 2, \ldots \,. \tag{42}$$

The critical number of pages c^* is now given as a function of c and ρ by the following theorem.

Theorem 1. *It holds that*

$$c^* \;=\; m \,, \quad \text{for} \;\; \rho \in [\rho_m, \rho_{m+1}) \,, \quad m = 0, 1, \ldots, c-2 \,, \tag{43}$$

with ρ_m *given by*

$$\rho_m \;=\; \frac{c-m}{c\, S(m+1, c)} \,, \quad \text{for} \;\; m = 0, 1, \ldots \,. \tag{44}$$

Proof: From the preceding, combining (41) and (42), and given that $\rho_{c-1} = 1$, yields (43)– (44). □

Corollary 1. For a given ρ*, as* c *increases, the normalized critical number of pages* c^*/c *increases and approaches* c^*_∞*, which satisfies the following relation:*

$$\rho \;=\; \frac{1 - c^*_\infty}{\log\left(\dfrac{1}{c^*_\infty}\right)} \,, \quad \forall\, \rho \in [0, 1) \,, \tag{45}$$

where

$$c^*_\infty \;\triangleq\; \lim_{c \to \infty} \frac{c^*}{c} \,. \tag{46}$$

Proof: From (43), it follows that c^*/c corresponds to m/c. Let us define the variable $x = m/c$, and replace the variable m in (42) with $x\,c$. Then ρ is given as a function of x by

$$\rho(x) \;=\; \lim_{c\to\infty} \frac{c - x\,c}{c\,S(x\,c+1,c)} \;=\; \lim_{c\to\infty} \frac{1-x}{S(x\,c+1,c)}\;, \quad \forall\, x \in [0,1]\;. \tag{47}$$

By recalling that for large values of c and n it holds that

$$S(n,c) \;=\; \sum_{j=n}^{c} \frac{1}{j} = \log\left(\frac{c}{n-1}\right)\;, \tag{48}$$

(47) yields

$$\rho(x) \;=\; \frac{1-x}{\log\left(\dfrac{1}{x}\right)}\;, \quad \forall\, x \in [0,1]\;. \tag{49}$$

□

The normalized average number of relocated pages v^* is now readily obtained by the following theorem.

Theorem 2. *It holds that*

$$v^* \;=\; \begin{cases} 0\;, & \text{for } \;\rho \le \rho_0 \\ \dfrac{(c^*+1)\,\{[1+S(c^*+2,c)]\,\rho - 1\}}{c\,\rho - (c^*+1)}\;, & \text{for } \;\rho > \rho_0\;. \end{cases} \tag{50}$$

Proof: For $\rho \le \rho_0$, according to (40), $\bar{V} = 0$. For $\rho > \rho_0$, c^* is obtained from (43) by identifying the interval $[\rho_{c^*}, \rho_{c^*+1})$ in which ρ lies. Substituting (39) into (25), and using (11) yields (50). □

Corollary 2. For a given ρ, as c increases, the normalized average number of relocated pages v^* increases and approaches v^*_∞, which satisfies the following relation

$$\rho \;=\; \frac{1-v^*_\infty}{\log\left(\dfrac{1}{v^*_\infty}\right)}\;, \quad \forall\, \rho \in [0,1)\;. \tag{51}$$

where

$$v^*_\infty \;\triangleq\; \lim_{c\to\infty} v^*\;. \tag{52}$$

Proof: From (11) and (25), it follows that

$$v^* \;=\; \frac{\bar{V}}{c} \;=\; \frac{c^*+1-q}{c}\;, \tag{53}$$

which implies that

$$v^*_\infty \;=\; \lim_{c\to\infty} v^* \;=\; \lim_{c\to\infty} \frac{c^*+1-q}{c} \;=\; \lim_{c\to\infty} \frac{c^*}{c} \;=\; c^*_\infty\;. \tag{54}$$

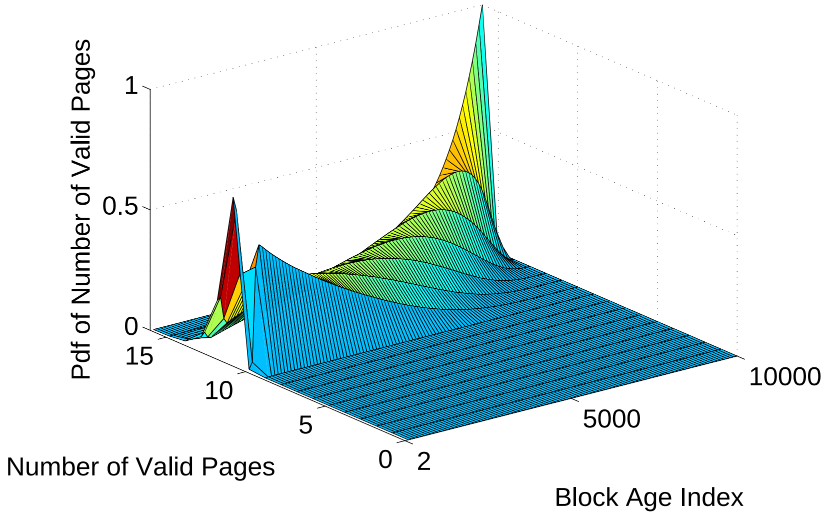

Figure 1: Distribution of the number of valid pages as a function of the block age index for $c = 16$, $b = 10,000$, and $\rho = 0.8$.

Combining (45) and (54) yields (51). □

The relation between v_∞^* and ρ given by (51) is in agreement with the one derived in [5].

5. Numerical Results

Here we study the performance of the greedy garbage-collection scheme and assess its impact on the write amplification. To confirm the theoretical results derived, we also developed a simulation model. Simulations were run until the measures of interest were observed to stabilize, indicating that the steady state had been reached. To accelerate the convergence of these measures, measurements taken during an initial warm-up phase were discarded. All simulation runs indicated that the the greedy policy resulted in a good degree of wear leveling as blocks were evenly erased and recycled, which implies that there was an even wear across all blocks.

We proceed by considering a system with $c = 16$, $b = 10,000$, and $\rho = 0.8$. Figure 1 shows the distribution of the number of valid pages as a function of the block age index, as defined in Section 3.1, which indicates the order according to which blocks were written. Note that the block with an age index of $b = 10,000$, corresponds to the most recently written block which, according to (17), contains $c = 16$ valid pages. This results in a spike with its height equal to one for a block age index equal to 10,000 and a number of 16 valid pages. As blocks age, and their block age index reduces, the corresponding number of valid pages also reduces. Figure 1 also shows that there are practically no blocks

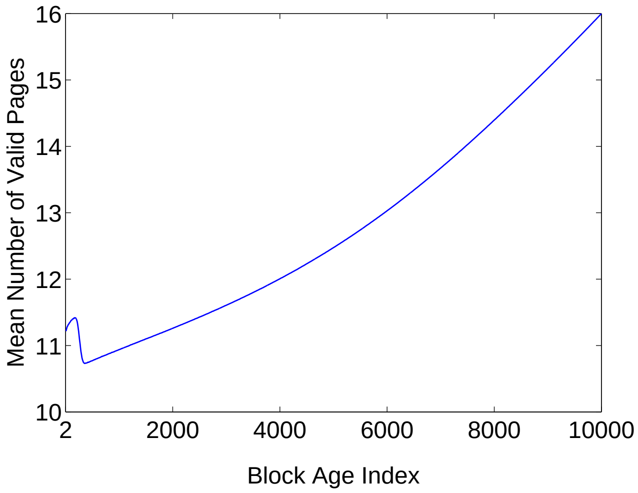

Figure 2: Mean number of valid pages as a function of the block age index for $c = 16$, $b = 10,000$, and $\rho = 0.8$.

with 9 or fewer valid pages, which confirms the validity of Eqs. (18) and (19) with the critical number of pages being equal to $c^* = 9$. From (42), it follows that $\rho_9 = 0.79 < \rho < 0.83 = \rho_{10}$, which, according to (43), implies that $c^* = 9$. Consequently, the analytical value of c^* is in agreement with the simulation. Figure 2 shows the mean number of valid pages as a function of the block age index. As previously mentioned, a block with an age index equal to $b = 10,000$ contains $c = 16$ valid pages. As blocks age, and their corresponding block age index reduces, their mean number of valid pages also reduces. However, those blocks with a relatively larger number of valid pages are not selected by the garbage-collection process, and therefore they become the oldest, that is, their block age index becomes small. This is illustrated in Figure 2 where for blocks with small values of age index, the corresponding mean number of valid pages is relatively large.

Figure 3 shows the distribution of the number of valid pages over all blocks at the times when garbage collection takes place, which also represents the steady-state probability p_j that a randomly selected block contains j valid pages. The square symbols correspond to the analytical results for infinitely large b, obtained by using (35), and are in good agreement with the simulation results (indicated by the circles), confirming the fact that there are practically no blocks containing $c^* = 9$ or fewer valid pages.

The distribution of the number of relocated pages, V, is plotted in Figure 4. The square symbols correspond to the analytical results derived by using (24) with $c^* = 9$ (given that $\rho_9 = 0.79 < \rho = 0.8 < \rho_{10} = 0.83$) and $q = 0.77$ as obtained by (39). These results are in good agreement with the simulation results (indicated by the circles) confirming that 77% of the garbage-collected blocks contain $c^* = 9$ valid pages and the remaining 23% of the garbage-collected

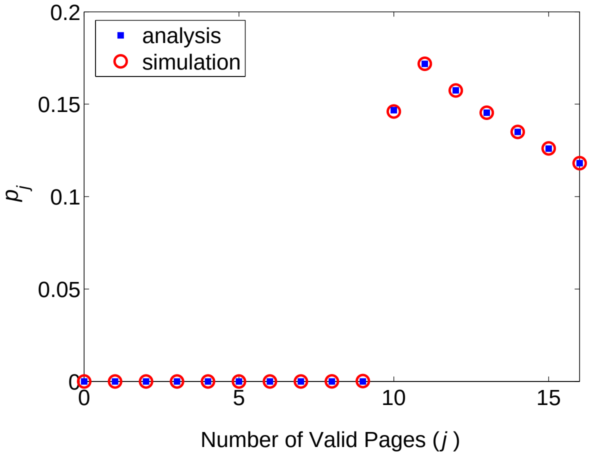

Figure 3: Distribution p_j of the number valid pages over all blocks (analysis + simulation) for $c = 16$, $b = 10,000$, and $\rho = 0.8$.

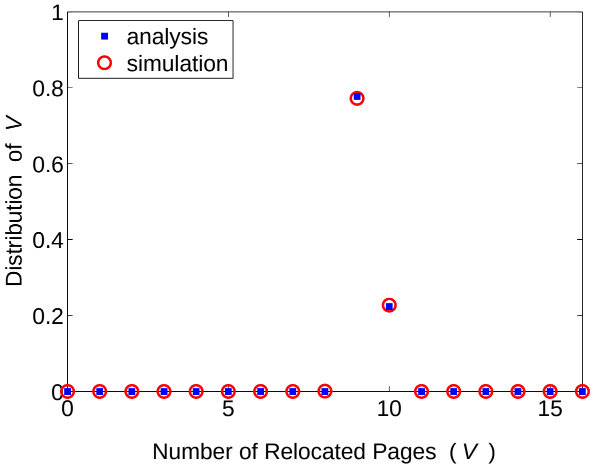

Figure 4: Distribution of V (analysis + simulation) for $c = 16$, $b = 10,000$, and $\rho = 0.8$.

blocks contain $c^* + 1 = 10$ valid pages.

We proceed by presenting analytical results for the various measures and for the entire range of ρ. Figure 5 plots the normalized critical number of pages, c^*/c, as a function of the system occupancy ρ and the number of pages per block c using (43). Note that c^* increases as ρ increases. It also increases, as c increases, and approaches a continuous curve given by (45).

The normalized average number of relocated pages, v^*, is plotted in Figure 6 as a function of the system occupancy ρ for various values of c using (50). As expected, v^* increases as ρ increases. Interestingly, the curves are continuous,

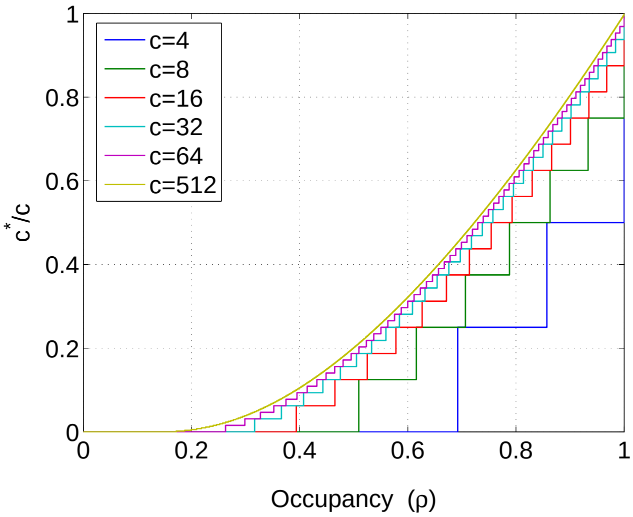

Figure 5: Normalized critical number of pages c^*/c as a function of ρ for $c = 4, 8, 16, 32, 64$, and 512.

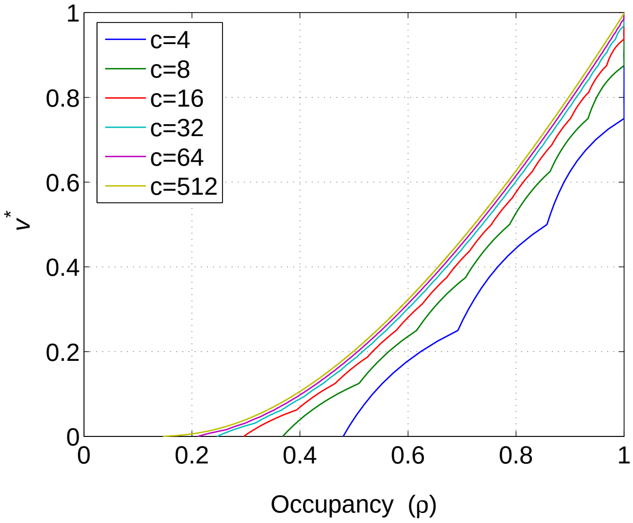

Figure 6: Normalized average number of relocated pages v^* as a function of ρ for $c = 4, 8, 16, 32, 64$, and 512.

but not smooth. The discontinuities occur precisely at the $\{\rho_m\}$ occupancy points at which c^* increases by one. Note that as c increases, v^* approaches a continuous smooth curve given by (51).

To verify the analytical results, we also ran simulations with $b = 1000$. The simulation results, indicated by the circles in Figure 7, reveal that for all values

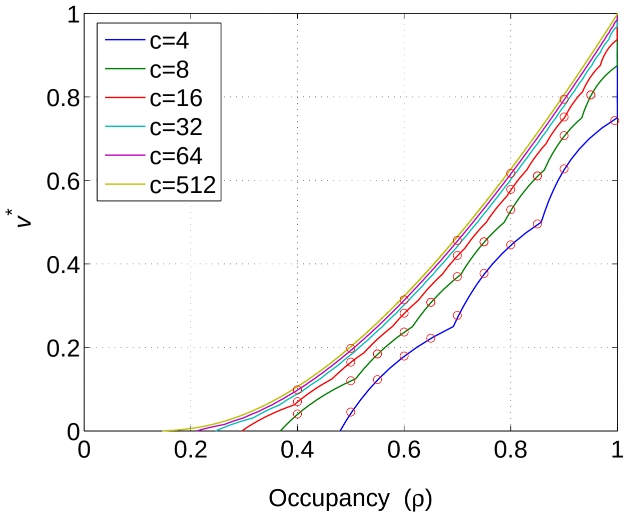

Figure 7: Normalized average number of relocated pages v^* (analysis + simulation with $b = 1000$) as a function of ρ for $c = 4, 8, 16, 32, 64$, and 512.

of occupancy and all numbers of pages considered, the analytical results are in excellent agreement with the simulated ones. Interestingly, they also agree for values for which the condition $u \gg c$ no longer holds. For example, for $c = 512$ and $\rho = 0.4$, we get $c^* = 54$, $\bar{V} = 54.36$, but it holds that $u = 400 < 512 = c$. In this case, however, the simulation results revealed that the number of relocated pages V is no longer either 54 or 55, but it can also be 53 or 56, such that $\bar{V} = 54.36$. Consequently, the condition $u \gg c$ is necessary and sufficient for the distribution of V to be bimodal, but this condition is not necessary for the results regarding $\bar{V}$ and v^* to be accurate.

The write amplification A is subsequently obtained by making use of (10), and plotted in Figure 8 as a function of the system occupancy ρ for various values of c. Interestingly, for small values of c, A increases almost linearly in the $[\rho_m, \rho_{m+1}]$ intervals. As can be seen from Figures 7 and 8, the average number of relocated pages, and therefore the write amplification, is less for smaller values of c. This is because operating with small blocks increases the likelihood that the garbage-collection process finds blocks for recycling that do not contain any valid pages. For instance, for $c = 1$, and for any system occupancy ρ, there always exist such blocks, such that the write amplification is equal to zero. However, there is a tradeoff between small and large block sizes. On the one hand, small values of c reduce both the write amplification and the blocking time due to an erase operation, but result on the other hand in an increased rate of block erasures, which in turn affects the performance of the device. Conversely, large values of c on the one hand reduce the rate of block erasures, and therefore improve the performance of the device, but on the other hand increase the write amplification and the blocking time due to an erase

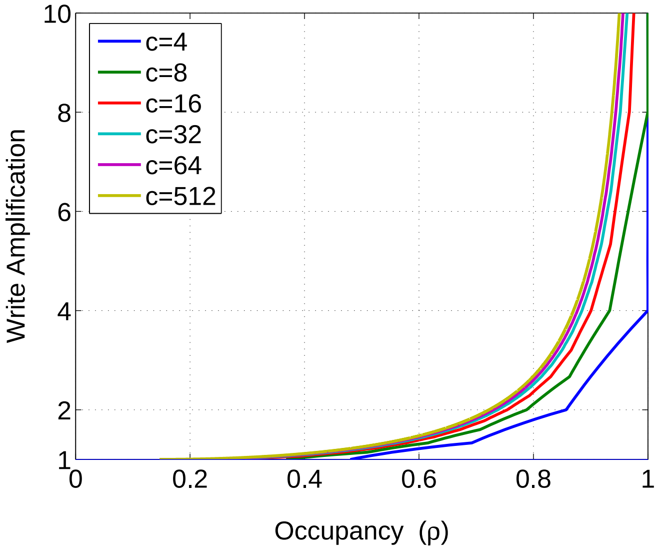

Figure 8: Write amplification A as a function of ρ for $c = 4, 8, 16, 32, 64$, and 512.

operation. A reasonable compromise is achieved by typically selecting $c = 64$. Thus, today's SSDs are comprised of blocks containing 64 pages of 4 KB each.

6. Conclusions

Today's data storage systems are increasingly adopting flash-based solid-state drives (SSD), in which, similarly to the log-structured file systems, new data is written out-of-place. The space occupied by the invalidated data is reclaimed by the garbage-collection process, which involves additional write operations that result in write amplification. The effect of the greedy garbage-collection scheme on the write amplification was assessed analytically for large flash-memory systems. Closed-form expressions were derived for the number of relocated pages and the write amplification.

Our theoretical results demonstrate that as the number of pages contained in a block increases, the write amplification increases and approaches an upper bound. Also, as the system occupancy increases, the write amplification increases. We find that the number of free pages reclaimed by the greedy garbage-collection mechanism after each block recycling takes one of two successive values, which provides a quasi-deterministic performance guarantee. Our simulation results confirm the analytical findings. They also show that the greedy garbage-collection mechanism inherently provides a good degree of wear leveling.

References

[1] J. Brewer, M. Gill, (ed.), Nonvolatile Memory Technologies with Emphasis on Flash: A Comprehensive Guide to Understanding and Using Flash Memory Devices, Wiley-IEEE Press, 2008.

[2] N. Agrawal, V. Prabhakaran, T. Wobber, J. D. Davis, M. Manasse, R. Panigrahy, Design tradeoffs for SSD performance, in: Proceedings of the 6th USENIX Annual Technical Conference (ATC) (Boston, MA), 2008, pp. 57–70.

[3] M. Rosenblum, J. K. Ousterhout, The design and implementation of a log-structured file system, ACM Trans. Comput. Syst. 10 (1) (1992) 26–52.

[4] J. Menon, L. Stockmeyer, An age-threshold algorithm for garbage collection in log-structured arrays and file systems, High Performance Computing Systems and Applications (1998) 119–132.

[5] J. Menon, A performance comparison of RAID-5 and log-structured arrays, in: Proceedings of the 4th International Symposium on High Performance Distributed Computing (HPDC) (Charlottesville, VA), 1995, pp. 167–178.

[6] X.-Y. Hu, E. Eleftheriou, R. Haas, I. Iliadis, R. Pletka, Write amplification analysis in flash-based solid state drives, in: Proceedings of the Israeli Experimental Systems Conference (SYSTOR) (Haifa, Israel), 2009, pp. 1–9.

[7] W. Bux, Performance evaluation of the write operation in flash-based solid-state drives, IBM Research Report, RZ 3757, IBM (Nov. 2009).

[8] L.-P. Chang, T.-W. Kuo, S.-W. Lo, Real-time garbage collection for flash-memory storage systems of real-time embedded systems, IEEE Trans. Embed. Comput. Syst. 3 (4) (2004) 837–863.