Using Bulk Arrivals to Model I/O Request Response Time Distributions in Zoned Disks and RAID systems

A.S. Lebrecht, N.J. Dingle, P.G. Harrison, W.J. Knottenbelt and S. Zertal

Department of Computing, Imperial College London, 180 Queen's Gate, London SW7 2BZ, United Kingdom
{asl102,njd200,pgh,wjk,szertal}@doc.ic.ac.uk

ABSTRACT

Useful analytical models of storage system performance must support the characteristics exhibited by real I/O workloads. Two essential features are the ability to cater for bursty arrival streams and to support a given distribution of I/O request size. This paper develops and applies the theory of bulk arrivals in queueing networks to support these phenomena in models of I/O request response time in zoned disks and RAID systems, with a specific focus on RAID levels 01 and 5. We represent a single disk as an $M^X/G/1$ queue, and a RAID system as a fork-join queueing network of $M^X/G/1$ queues. We find the response time distribution for a randomly placed request within a random bulk arrival. We also use the fact that the response time of a random request with size sampled from some distribution will be the same as that of an entire batch whose size has the same distribution. In both cases, we validate our models against measurements from a zoned disk drive and a RAID platform.

Keywords

queueing theory, storage systems

1. INTRODUCTION

Despite the current economic downturn, demand for disk storage continues its unrelenting rise. Indeed, the IDC forecasts that shipped disk storage capacity will increase at a compound annual growth rate of over 38% for the next three years [21]. The efficient operation of public and private enterprises worldwide remains critically dependent on reliable, high performance storage. The detailed understanding of disk storage system performance is therefore crucial in determining whether storage infrastructures will deliver their required quality of service.

It is imperative that models of storage system performance should be capable of reflecting in their inputs the features found in real I/O workloads. Over the last decade, there have been many studies of storage system I/O traces (e.g. [8, 27, 19, 20, 23]). These consistently show that real-life arrivals to storage systems exhibit burstiness and a variety of request size distributions.

While analytical models of disks and RAID systems provide a rapid and elegant means to make performance predictions, it is challenging to incorporate the features identified above into them. Some progress has been made on reflecting variable sized I/O requests [4, 18], but to the best of our knowledge all existing models of RAID assume a simple non-bursty Markovian I/O request arrival stream (e.g. [5, 11, 12, 13, 15, 25, 26]). We also note that prior to [12] and [13], all previous analytical studies focused only on mean I/O request response time; however, modern Service Level Agreements demand the ability to reason about higher moments and percentiles of response time. For certain highly constrained Markovian arrival streams, [12, 13] derive the full cumulative distribution function of I/O request response time, from which all of the previous measures can be easily obtained.

This paper makes contributions on both theoretical and application levels. Specifically, we develop novel analytical methods for calculating response times in $M/G/1$ queues with bulk arrivals (i.e. $M^X/G/1$ queues), as well as for approximating response times in fork-join networks of such queues. We then apply these in an analytical queueing network model of zoned disk drives and RAID systems to yield the full distribution of I/O request response time given a) arrival streams with burstiness characterised by bulk arrivals and b) arrival streams with an arbitrary distribution of request response size.

In the context of arrival streams with burstiness, we derive the response time distribution of a randomly-selected request within a batch. Previous work [2, 10] calculates only the queue length generating function of a single $M^X/G/1$ queue.

Our chosen method to model a workload of different-sized jobs is to define each job as a single bulk arrival, where the size distribution of the bulk arrivals is the same as the job size distribution. We then seek the response time distribution of an entire batch, rather than a random customer within a batch, which is the usual focus on work on bulk arrivals. To the best of our knowledge, only Nelson, Towsley and Tantawi [18] have applied this perspective, in modelling the parallel processing of different sized requests in an $M^X/M/c$ queue. Here we consider fork-join networks of $M^X/G/1$ queues since it is well known that disk drives do not have Markovian service times, and a fork-join structure accurately reflects the operation of RAID systems [15].

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
VALUETOOLS 2009, October 20–22, 2009 – Pisa, Italy
Copyright 2009 ICST 978–963–9799–70–7/00/0004 $5.00.

Our analysis supports RAID levels 01 and 5. We note that our RAID 5 write model is more general than existing analytical models which presume RAID 5 write operations consist of a fixed number of subtasks in a job [11, 13] or a variable number that never exceeds a full stripe [4]. Furthermore, all these models assume that all requests are performed without skewing (i.e. assuming that requests are stripe-aligned) which is not in general the case in reality. Our model relaxes both these restrictions.

The remainder of this paper is organised as follows. Section 2 describes relevant background theory, before Section 3 presents our development of bulk arrival theory to permit the analysis of $M^X/G/1$ queues for full response time distributions. We extend this approach to allow the calculation of such distributions in fork-join networks composed of several of these queues, thus enabling us to model single disks and RAID systems with bursty arrivals. Section 4 presents our approach for calculating the response time distribution of an entire batch of arrivals, using $M^X/G/1$ models of single disks and fork-join network models of RAID systems. This gives us the ability to represent variable-sized I/O requests arriving to an $M/G/1$ queue. Section 5 then validates our methods against measurements from real devices. Section 6 concludes and considers directions for future work.

2. BACKGROUND

We develop our RAID model in a bottom-up hierarchical fashion. A disk drive is modelled as a single $M/G/1$ queue, and a RAID system is abstracted as a fork-join queueing network [3].

2.1 Disk Model

The service time density of an access to a random location on a single disk drive is the convolution of the seek time, rotational latency and data transfer time probability density functions. An important advance in disk drive technology that needs to be taken into account is that modern disks are *zoned*, with more sectors on the outer tracks than inner tracks. A random request is thus more likely to access a sector on an outer track. Similarly, zoning means that it is faster to transfer data on a track close to the circumference than the centre of the disk. The seek time and data transfer models must take these factors into account.

Our zoned disk model uses the seek time and rotational latency probability distributions defined in [29] and the data transfer time distribution from [13]. We denote the random variables of seek time, rotational latency and single block transfer time as S, R and T respectively.

2.2 Fork-Join Model

In an N-queue fork-join network (see Figure 1) each incoming request or job is split into N subtasks at the fork point. Each of these subtasks queues for service at a parallel service node before joining a queue for the join point. When all N subtasks in the job are at the head of their respective join queues, they rejoin (synchronise) at the join point.

It is difficult to model job response times in a fork-join synchronisation analytically. Indeed, exact analytical results only exist for the mean response time of a two server system consisting of homogeneous $M/M/1$ queues [17]. Approximations for mean response times for $M/M/1$ and $M/G/1$ fork-join queues are more abundant [17, 24, 25] but such results do not permit higher moments or full response time distributions to be calculated. Here, as in [13], we use an approach based on the maximum order statistic [7, 14] to derive an approximation to the cumulative distribution function of a fork-join queue's response time. That is, if a single queue has a response time distribution $F(x)$, then the response time distribution of a fork-join queueing network of n identical queues is approximated as $F(x)^n$.

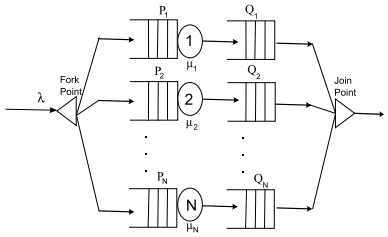


Figure 1: Fork-join queueing model

2.3 RAID Model

A fork-join queue does not model all the intricacies of a RAID system. The fork-join analysis described above can calculate the response time cdf for read or write requests to an n-disk RAID system in which each request consists of a multiple of n blocks. However, not every I/O request leads to an access to all disks, being influenced by I/O request size and type, and also by RAID level. We adapt the notation used in [13], in which the fork-join approximation was tailored to model I/O operations on mirrored stripes (RAID 01) and distributed parity (RAID 5). For simplicity, we always assume n homogeneous disks in the array; we also assume an arrival rate of λ logical I/O requests per ms. Let $W_d(t, \gamma, \mu)$ define the cumulative distribution function (cdf) of the response time of a single $M/G/1$ queue (disk), where γ is the arrival rate at an individual disk and μ is the mean service rate of a single disk. Then, for example, the response time distribution of a constant-sized read request, b, on RAID 01 is:

$$W_{\text{read}}(t) = \begin{cases} \left(W_d\left(t, \frac{\lambda b}{n}, \frac{1}{E[R]+E[S]+E[T]}\right)\right)^b & \text{if } b < n \\ \left(W_d\left(t, \lambda, \frac{1}{E[R]+E[S]+\frac{b}{n}E[T]}\right)\right)^n & \text{otherwise} \end{cases}$$

Interested readers are invited to consult [13] for a full description of this model.

3. WORKLOADS WITH BULK ARRIVALS

In this section we derive the response time distribution for a random job in an $M^X/G/1$ queue. Our approach is inspired by Harrison's derivation of the Laplace-Stieltjes Transform (LST) of job response time in an $M/G/1$ queue using conditional probability [9].

3.1 Job Response Time in a Single Queue

Figure 2 shows the state of an $M^X/G/1$ queue at the arrival instant of a randomly chosen job, given that the queue is not empty at the arrival instant. At the arrival instant, a batch C is currently in service. This batch has completed Ums of service (the backward recurrence time [6]) and has Vms of service remaining (the forward recurrence time).

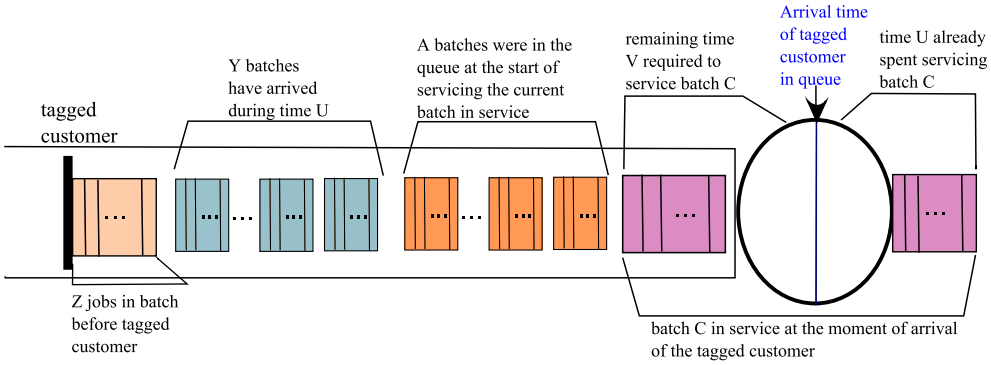


Figure 2: The queue at the arrival instant of a tagged customer

When batch C started service, there were A batches queueing behind it. During time U, a further Y batches joined the queue. Within the arriving batch, there are Z jobs (the backward recurrence size) ahead of the tagged customer. U and V are continuous random variables, and A, Y and Z are discrete random variables.

If B denotes the random variable describing batch size, and X the service time of a single job, then the LST of the service time of an entire batch $X_B^*(\theta)$ can be defined as:

$$\begin{aligned} X_B^*(\theta) &= E[E[e^{-\theta(X_1+X_2+\ldots+X_B)}] \\ &= E[(X^*(\theta))^B] \\ &= G_B(X^*(\theta)) \end{aligned}$$

where $G_B(z)$ is the probability generating function of B. It is straightforward to show $E[X_B] = E[X]E[B]$.

The queueing time of the tagged customer, denoted by random variable Q, is the sum of the service times of all customers ahead (including the remaining service time of the job in service), as follows:

$$\begin{aligned} &E[e^{-\theta Q} \mid Q > 0] = \\ &E[e^{-\theta(V+X_{B_1}+\ldots+X_{B_A}+X_{B_1}+\ldots+X_{B_Y}+X_1+\ldots+X_Z)}|U,V,Y,Z,A] \\ &\qquad = X^*(\theta)^Z G_B(X^*(\theta))^A G_B(X^*(\theta))^Y e^{-\theta V} \end{aligned}$$

Deconditioning on A and Z (which are independent of V, Y and each other), and Y (which can be expressed in terms of U given that the number of batches arriving has a Poisson distribution),

$$E[e^{-\theta Q} \mid Q > 0] = P(\theta)E[e^{-\theta V} e^{-\lambda(1-(G_B(X^*(\theta))))U} \mid U, V]$$

where $P(\theta) = G_Z(X^*(\theta))G_A(G_B(X^*(\theta)))$. Here $G_Z(z)$ is the generating function of the discrete backward recurrence time (see Appendix A). $G_A(z)$ is the generating function of queue length at the beginning of service of the first customer in a batch. By the random observer principle, at equilibrium A is equivalent to N, the number of customers queueing immediately after the start of a batch service. $G_N(z)$ is a well known result [10]:

$$G_A(z) = G_N(z) = \frac{(1-\rho)(1-z)}{G_B(X^*(\lambda(1-z)) - z}$$

The joint density of forward and backward recurrence times U and V at a point (u, v) is $\frac{1}{E[X_B]} f_{X_B}(u+v)$ [10] where $f_{X_B}(t)$ is the pdf of batch service time X_B. Thus, deconditioning further,

$$\begin{aligned} &E[e^{-\theta Q} \mid Q > 0] = \\ &\frac{P(\theta)}{E[X]E[B]} \int_0^\infty \int_0^\infty e^{-\theta v} e^{-\lambda(1-(G_B(X^*(\theta))))u} f_{X_B}(u+v)\,du\,dv \\ &\qquad = \frac{P(\theta)(G_B(X^*(\lambda(1-G_B(X^*(\theta)))))) - G_B(X^*(\theta)))}{E[X]E[B](\theta - \lambda(1-(G_B(X^*(\theta)))))} \end{aligned}$$

If a batch arrives to an empty queue, then the queueing time for a randomly selected job in the batch is the time to service all the jobs ahead of it in the batch; thus

$$E[e^{-\theta Q} \mid Q = 0] = G_Z(X^*(\theta))$$

Considering both cases (empty and non-empty queues), and given queue utilisation ρ, the LST of queueing time is:

$$\begin{aligned} Q^*(\theta) &= (1-\rho)G_Z(X^*(\theta)) + \rho E[e^{-\theta Q}|Q > 0] \\ &= \frac{(1-\rho)\theta G_Z(X^*(\theta))}{\theta - \lambda(1-(G_B(X^*(\theta))))} \end{aligned}$$

Hence the response time LST for a randomly placed customer in a batch is

$$W^*(\theta) = Q^*(\theta)X^*(\theta). \tag{1}$$

The response time distribution is obtained by numerically inverting $W^*(\theta)$ [1].

In our context of zoned disk modelling with bulk arrivals, we represent a single disk as an $M^X/G/1$ queue with the service time distribution referred to in Section 2.1. These queues also form the basic components of our fork-join model for RAID systems.

3.2 RAID 01

RAID 01 is the striped mirrored RAID level, where disks are divided into 2 identical sets (native, mirror). A RAID 01 read access can be executed on either the native or the mirror copy while the write access must be done on both.

It is not straightforward to extend the single disk model for batch arrivals to RAID 01. Our fork-join approximation assumes independence of response times for each disk; however, in the case of batch arrivals to RAID, although the service time distributions on the different disks can reasonably be assumed to be independent, under the assumption that all operations are full-stripe accesses, each queue will receive the same number of jobs per batch and hence the queueing time will have a high level of dependency.

Therefore we use a new method to calculate the response time distribution. The model results in a single queue whose

service time distribution is calculated as the distribution of the maximum service time across all the disks in the array. This assumption of dependence of queueing (but not service) times is not exact, but approximates the reality that queueing times of requests to each disk in a RAID system will be highly correlated.

We begin by finding $F_X(t)$, the service time distribution of a single zoned disk, defined as the sum of seek time, rotational latency and data transfer time. The intricacies of zoning mean that this distribution cannot be found analytically, and must instead be calculated numerically by inversion of its LST or by convolution. It is then possible to find the distribution of maximum service time across n disks using the maximum order statistic, i.e. $F_X(t)^n$.

Equation 1 (from which we will derive our response time distribution) requires the LST of the maximum service time $X^*(\theta)$. Numerical calculation of this LST is theoretically possible, but in practice requires an infeasible amount of computation. As a means to more efficiently and elegantly obtain the LST, we proceed by fitting a logistic function:

$$f(t) = \frac{1}{1 + e^{a-bt}}$$

to the distribution of maximum service time $F_X(t)^n$. The fitting can be accomplished by using a nonlinear least-squares Marquardt-Levenberg algorithm [16]. The LST of the logistic function is then:

$$X^*(\theta) = \text{Hypergeometric2F1}[1, s/b, (b+s)/b, -e^a]$$

where Hypergeometric2F1 is the hypergeometric function ${}_2F_1(a, b, c, z)$ which is the solution for y of the hypergeometric differential equation [28]:

$$z(1-z)y'' + [c - (a+b+1)z]y' - aby = 0$$

Substitution of $X^*(\theta)$ into Equation 1 then gives a readily-invertible expression for the distribution of response time.

4. WORKLOADS WITH DIFFERENT SIZED ARRIVALS

We now consider a different application of bulk arrival theory, namely to support the modelling of Markovian (i.e. without bulk arrivals) arrival streams with different size requests. In the following, I/O requests are equivalent to batches, and subtasks making up a request are equivalent to jobs within a batch. We assume all requests are random disk accesses. Thus, the first subtask within a request will have a service time that includes time to seek and rotate to the desired position on the disk. The remaining subtasks will be sequential.

4.1 Single Disk

The number of subtasks in a request, B, can be described by a discrete probability density function, $f_B(x)$ with probability generating function $G_B(x)$. The first subtask in the batch will have a service time $X_{RAND} = S + R + T$. The remaining subtasks are sequential and hence have service time $X_{SEQ} = T$. These service time random variables have corresponding LSTs $X^*_{RAND}(\theta)$ and $X^*_{SEQ}(\theta)$. As before, we assume all service times are independent. Then, if we define X_B as the random variable of the service time of all the subtasks in a single request, the corresponding LST can be calculated as follows:

$$X^*_B(\theta) = E[E[e^{-\theta(X_{RAND} + X_{SEQ_1} + \ldots + X_{SEQ_{B-1}})}|B]] \qquad (2)$$

There is always one fewer sequential subtask than there are subtasks in a batch. The number of sequential subtasks has a probability density function $f_{B-1}(x) = f_B(x+1), x = 0, 1, 2, \ldots$. Since X_{RAND} and X_{SEQ} are independent,

$$\begin{aligned} X^*_B(\theta) &= E[E[e^{-\theta X_{SEQ}}]^{(B-1)} E[e^{-\theta X_{RAND}}]] \\ &= E[(X^*_{SEQ}(\theta))^{(B-1)} X^*_{RAND}(\theta)] \\ &= G_{B-1}(X^*_{SEQ}(\theta)) X^*_{RAND}(\theta) \end{aligned}$$

where $G_{B-1}(z) = \sum_{i=0}^{\infty} f_B(i+1) z^i$. This LST is substituted into the Pollaczek-Khintchine transform equation [10] as the LST of service time; hence the response time LST is:

$$W^*_d(\theta) = \frac{(1-\rho)\theta X^*_B(\theta)}{\lambda X^*_B(\theta) - \lambda + \theta} \qquad (3)$$

where $\rho = \lambda E[X_B]$. The mean job service time, $E[X_B]$ can be calculated from $X^*_B(\theta)$, yielding:

$$\begin{aligned} E[X_B] &= E[X_{RAND}] + E[X_{SEQ}]E[B-1] \\ &= E[S] + E[R] + E[B]E[T] \end{aligned}$$

As before, $W^*_d(\theta)$ is easily numerically inverted to obtain the distribution of service time.

4.2 RAID 01

In RAID 01, subtasks from each request are striped across the disks. A read request stripes all its subtasks across the disks and a corresponding write request stripes double the number of subtasks. As a consequence of the striping, a single disk in a disk array will only see a fraction of the subtasks of a request. If we define B_R as the number of subtasks in a request on a single disk in the array and the mean number of subtasks in a request is $E[B]$, the mean request size per utilised disk, $E[B_R]$, is:

$$E[B_R] = \begin{cases} 1 & E[B] < n \\ \frac{E[B]}{n} & \text{otherwise} \end{cases}$$

Consequently, the number of disks in the array accessed by each request depends on the number of subtasks in a job and is therefore dependent on B. We note all disks will be accessed unless a job has fewer subtasks than there are disks. We define the parameter d_B as the mean number of disks in use:

$$d_B = n - \sum_{i=1}^{n-1} (n-i)\ f_B(i)$$

Similarly the arrival rate, γ, at each disk is dependent on the number of disks accessed.

$$\gamma = \frac{\lambda}{n} \left(p_{read} d_B + p_{write} d_{2B}\right)$$

The response time distribution of a read and write request on RAID 01 can now be defined as:

$$W_{\text{read}}(t) = \left(W_d\left(t, \gamma, \frac{1}{E[R] + E[S] + E[B_R]E[T]}\right)\right)^{d_B}$$

$$W_{\text{write}}(t) = \left(W_d\left(t, \gamma, \frac{1}{E[R] + E[S] + E[2B_R]E[T]}\right)\right)^{d_{2B}}$$

The single disk cdf $W_d(t, \gamma, \mu)$ is the numerical inversion of the LST in Equation 3 with $X^*_B(\theta)$ replaced by $X^*_{B_R}(\theta)$.

4.3 RAID 5

RAID 5 is block-interleaved distributed parity. The parity is defined as the XOR of all the data on a stripe.

Owing to the way in which RAID 5 distributes its parity blocks, RAID 5 reads can be modelled in a similar manner to RAID 01 reads [13].

RAID 5 write operations are significantly more complex due to the need to update parity blocks. If a full stripe is written, then all the new data is immediately available for parity calculation; however, if less than a full stripe is written, then the new parity can only be calculated with old data already written on the stripe. There are two ways to update the parity: a read-modify-write finds the XOR of the data and parity that are being overwritten with the new data; a read-reconstruct-write finds the XOR of the old data on the stripe that is not being overwritten with the new data. In order to calculate the parity the old data needed must be pre-read before the new data and parity can be written.

A randomly-sized RAID 5 write with a skew (i.e. not stripe-aligned) could consist of any or all of the following: a partial stripe write followed by a number of full stripe writes followed by another partial stripe write. We summarise the five possible procedures $(P_1, \ldots, P_5)$ for a RAID 5 write in Table 1. Here *Rand* is a random seek and transfer, *Seq* is a sequential operation and NOP represents no operation.

	P_1	P_2	P_3	P_4	P_5
pre-read	Rand	NOP	NOP	NOP	Rand
partial stripe write	Rand	NOP	NOP	NOP	Rand
full stripe	Seq	Rand	NOP	Rand	NOP
further full stripes	Seq	Seq	NOP	Seq	NOP
pre-read	Seq	Seq	Rand	NOP	Seq
partial stripe write	Rand	Rand	Rand	NOP	Rand

Table 1: Possible RAID 5 write procedures

Table 1 shows that there will never be more than three random accesses in a single RAID 5 write request. There will only be one random operation in the case that there are no partial stripes writes, namely that a job begins at the start of a stripe and that the number of subtasks in the job is divisible by $n-1$. However, our RAID model aims to find an average amount of accesses on a single disk and then apply it to all disks. This is more difficult in RAID 5, as the parity pre-reads and partial stripe writes are made only to certain disks in the array, depending on whether the parity calculation demands a read-modify-write or read-reconstruct-write. It is very likely, however, that no single disk will have more than two random accesses directed to it, since a partial stripe does not access all disks. Meanwhile, each disk will only have one random access per request if the request consists of a multiple of $n-1$ subtasks and the request starts at the beginning of a stripe (i.e. a full stripe write). Defining $f_{R5R}(x)$ as the probability of encountering x random subtasks in a request on a single disk, in terms of the job size probability distribution, $f_B(x)$:

$$f_{R5R}(x) = \begin{cases} \frac{1}{n-1}\sum_{i=1}^{\infty} f_B(i(n-1)) & x = 1 \\ 1 - \frac{1}{n-1}\sum_{i=1}^{\infty} f_B(i(n-1)) & x = 2 \\ 0 & \text{otherwise} \end{cases}$$

Similarly, defining $f_{R5S}(x)$ as the probability of encountering x sequential subtasks in a request on a single disk:

$$f_{R5S}(x) = \sum_{i=x(n-1)+1}^{(x+1)(n-1)} f_B(i)$$

In a similar way as for Equation 2, we derive $X_B^*(\theta)$ as:

$$X_B^*(\theta) = G_{R5S}(X_{RAND}^*(\theta))G_{R5S}(X_{SEQ}^*(\theta))$$

This can be used to calculate $E[X_B]$ straightforwardly.

In terms of the number of disks accessed, a RAID 5 write request accesses all disks unless it is a partial stripe. A small partial stripe write (read-modify-write), which occurs when $b < \frac{n-1}{2}$, involves accesses to $b+1$ disks (b data disks plus the parity disk). A large partial stripe write (read-reconstruct-write) involves accesses to all disks over two operations, and so on average accesses $\frac{n}{2}$ disks per operation. Hence:

$$d_B = \sum_{i=1}^{\lfloor \frac{n-1}{2} \rfloor} (i+1) f_B(i) + \frac{n}{2} \sum_{i=\lceil \frac{n-1}{2} \rceil}^{n-1} f_B(i) + n\left(1 - \sum_{i=1}^{n} f_B(i)\right)$$

5. VALIDATION

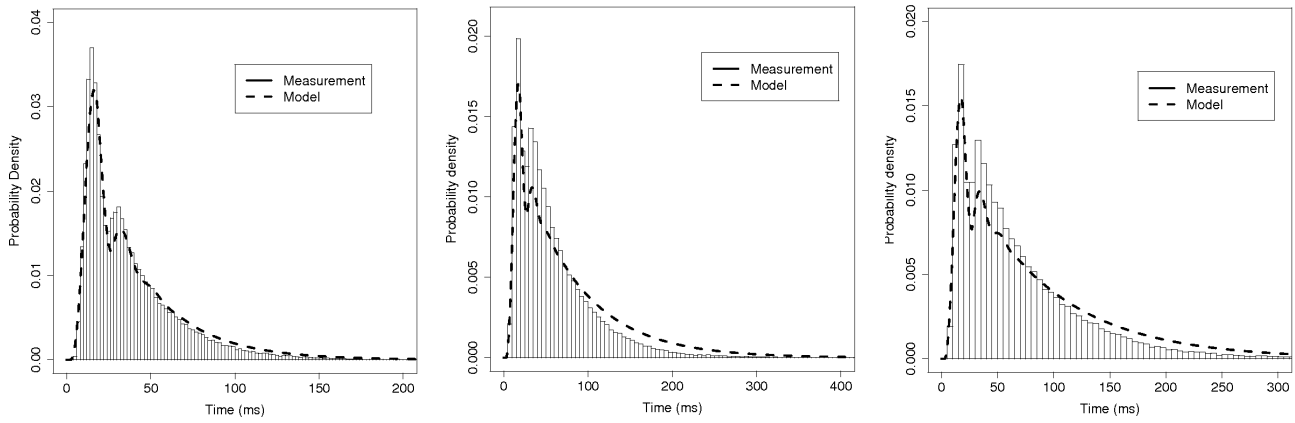
Our experimental platform consists of an Infortrend A16F-G2430 RAID system containing four Seagate ST3500630NS disks. Each disk has 60 801 cylinders. A sector is 512 bytes and we have approximated, based on measurements from the disk drive, that the time to write a single physical sector on the innermost and outermost tracks are 0.012064ms (t_{max}) and 0.005976ms (t_{min}) respectively. The stripe width on the array is configured as 128KB, which we define as the block size. Therefore there are 256 sectors per block. The time for a full disk revolution is 8.33ms. A track to track seek takes 0.8ms and a full-stroke seek requires 17ms for a read; the same measurements are 1ms and 18ms respectively for a write [22].

To obtain response time measurements from this system, we implemented a benchmarking program that issues read and write requests using a master process and multiple child processes. These child processes are responsible for issuing and timing I/O requests, leaving the master free to spawn further child processes without the need for it to wait for previously-issued operations to complete.

In order to validate the analytical model effectively, it was necessary to minimise the effects of buffering and caching as these are not currently represented in the model. We therefore disabled the RAID system's write-back cache, set the read-ahead buffer to 0 and opened the device with the `O_DIRECT` flag set. For each of the experiments presented below, 100 000 requests were issued. We choose to present all our results as probability density functions (pdfs); however, it would be possible to derive the cumulative distribution function, mean, variance and further moments of response time if required.

5.1 Bulk Arrivals

Figure 3 compares model predictions and measurement results for bulk arrivals to a single disk, with the number of requests in a batch generated using a geometric distribution. Results are presented for two different arrival rates ($\lambda = 0.01, 0.02$ requests per ms) and for arrival streams composed entirely of batches of either read or write requests. All requests within batches consist of one 128KB block and are to random locations. We observe good agreement between



(a) mean batch size = 2, reads, $\lambda = 0.01$ (b) mean batch size = 2, writes, $\lambda = 0.02$ (c) mean batch size = 3, writes, $\lambda = 0.01$

Figure 3: Response time pdf of model against measurement for requests on a single disk with batch arrivals.

model and measurement. The multiple peaks observed in both model and measurement arise from the variation of response time caused by the different possible batch sizes (with the most probable batch sizes yielding the highest peaks).

As described in Section 3.2, we can perform a least-squares fit of a logistic function to the (numerically calculated) distribution of maximum disk service time in order to more efficiently generate response time results for RAID 01 systems with batch arrivals. Figure 4 shows the logistic function fit for a 4-block read operation. The closeness of fit gives confidence in the accuracy of this approximation step.

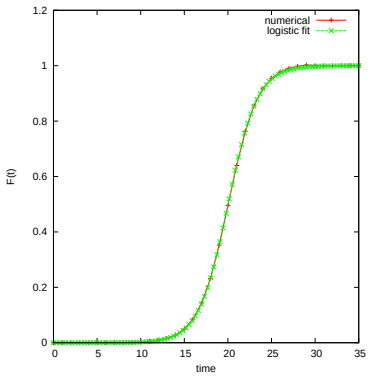


Figure 4: Logistic fit to maximum disk service time cdf

Figure 5 compares model predictions and measurement results for bulk arrivals to a four-disk RAID 01 system. As with the single disk, the number of requests in a batch is generated using a geometric distribution but results are presented for both read and write requests for a single arrival rate of $\lambda = 0.01$ requests per ms. Again, we observe good agreement between model and measurement. It is interesting to note that the model predicts more pronounced peaks than the measurements. We speculate that this is due to the nature of our maximum order statistic approximation for service time in a fork-join queue, which magnifies the sinusoidal behaviour of the single disk model, as well as our assumption of dependence of queueing times at each disk.

5.2 Variable Job Sizes

Figure 6 compares model predictions and measurement results for variable-sized arrivals to a single disk where the size of a request is generated using a geometric distribution. Results are presented for two different arrival rates ($\lambda = 0.01, 0.02$ requests per ms) and for arrival streams composed entirely of either read or write requests. In the case of read requests, we observe excellent agreement between model and measurement. In the case of write requests (Figures 6(c) and 6(f)) the agreement is less good, but is still reasonable and yields similar means and variances of response time. We speculate that the bimodal nature of the measurements may be due to some disk-specific write request handling behaviour that is not accounted for in the model.

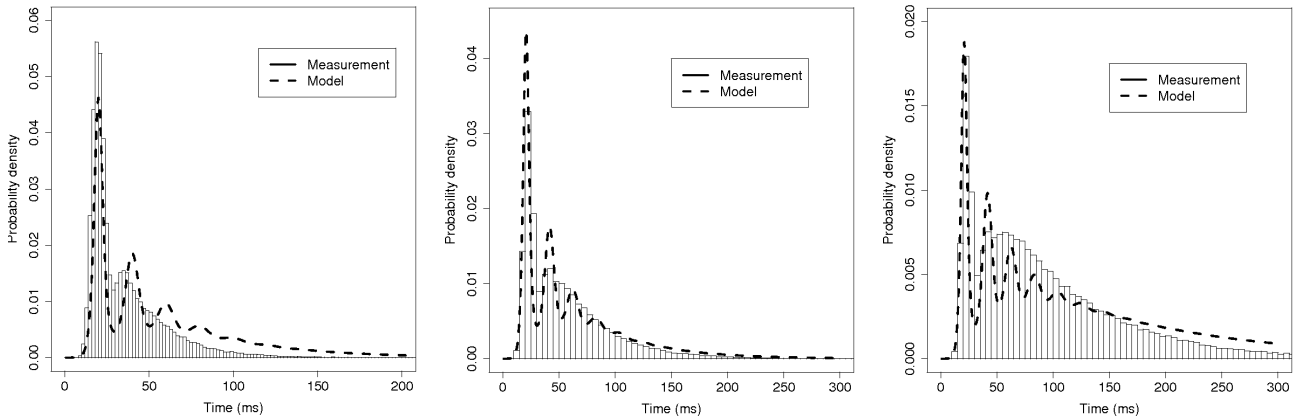
Figure 7 compares model predictions and measurement results for bulk arrivals to a four-disk RAID 01 system. As with the single disk, the size of request is generated using a geometric distribution. We observe reasonable agreement between model and measurement.

We note that all results thus far have been presented for arrival streams composed entirely of read or write requests. Our model, however, is capable of calculating results for arrival streams containing a mixture of both reads and writes. Figure 8 accordingly compares model predictions and measurement results in this case; we observe good agreement.

Finally, Figure 9 compares model predictions and measurement results for arrivals with geometrically-distributed sizes to a four-disk RAID 5 system. We observe excellent agreement for read requests. The fit for write requests is less exact due to the complicated nature of the pre-reads and parity updates that is approximated in our models.

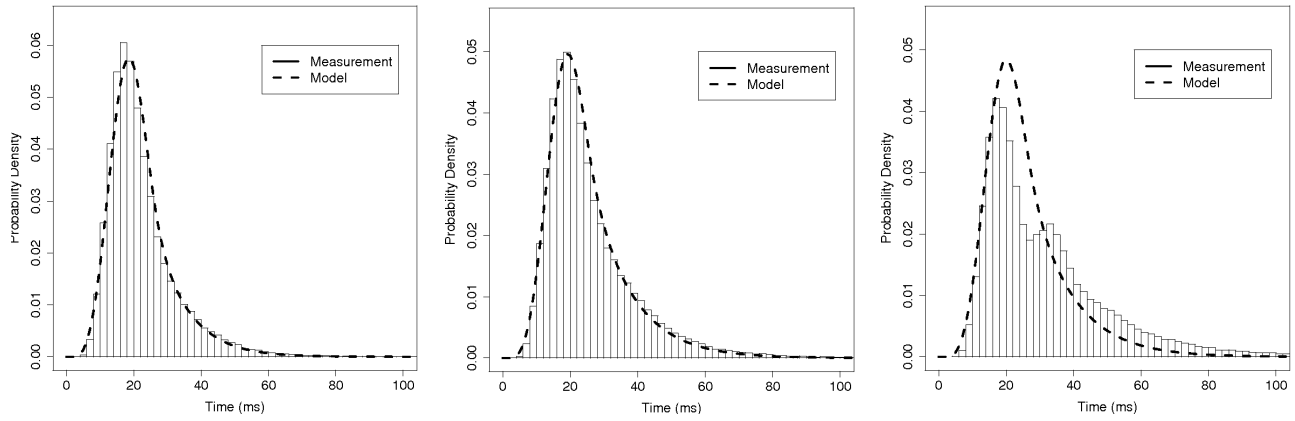
6. CONCLUSIONS AND FUTURE WORK

In this paper we have developed new analytical methods for response times of bulk arrivals in queueing networks, and we have applied these in predicting I/O request response time in zoned disks and RAID systems operating at levels 01 and 5. Our derivations for the distribution of I/O request response time given bursty and variable-sized arrival streams allow us to more accurately reflect workloads experienced by real-life storage systems within our models. The applicability and accuracy of our models is shown by our validation

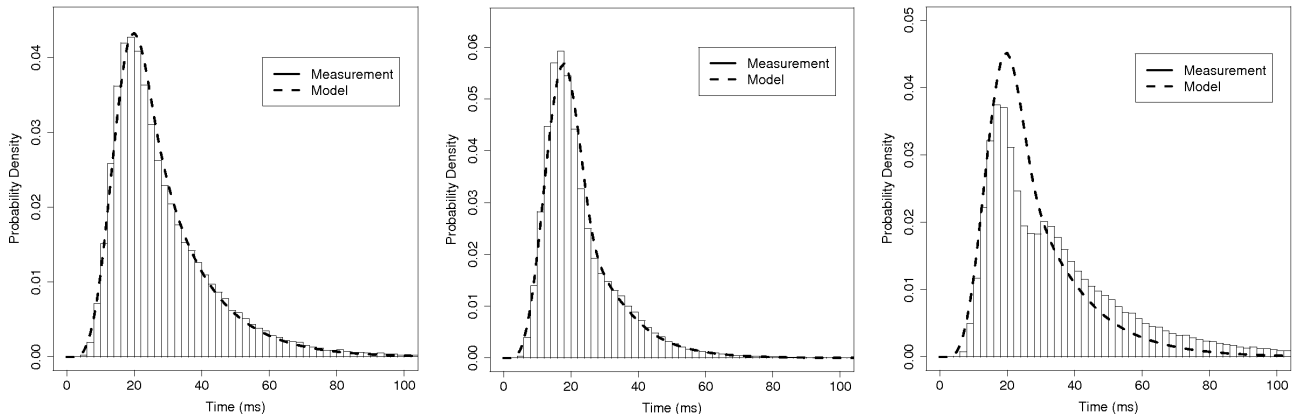


(a) mean batch size = 2, reads, $\lambda = 0.01$ (b) mean batch size = 2, writes, $\lambda = 0.01$ (c) mean batch size = 3, writes, $\lambda = 0.01$

Figure 5: Response time pdf of model against measurement for requests on a RAID 01 with batch arrivals.

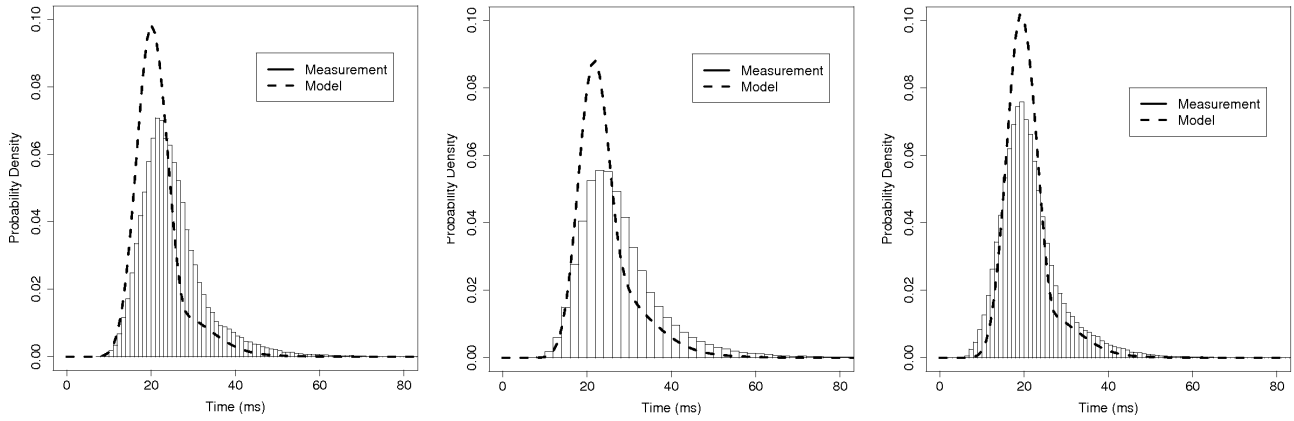


(a) mean job size = 3, reads, $\lambda = 0.01$ (b) mean job size = 4, reads, $\lambda = 0.01$ (c) mean job size = 4, writes, $\lambda = 0.01$



(d) mean job size = 5, reads, $\lambda = 0.01$ (e) mean job size = 2, reads, $\lambda = 0.02$ (f) mean job size = 3, writes, $\lambda = 0.02$

Figure 6: Response time pdf of model against measurement for requests on a single disk with different sized arrivals.



(a) mean job size = 2, writes, $\lambda = 0.01$ (b) mean job size = 3, writes, $\lambda = 0.01$ (c) mean job size = 4, reads, $\lambda = 0.01$

Figure 7: Response time pdf of model against measurement for requests on RAID 01 with different sized arrivals.

of model predictions against device measurements.

There are a number of potential avenues for future work. Firstly, our RAID 01 batch arrival model currently assumes full-stripe arrivals, but this restriction could be relaxed. Secondly, our validation experiments have used a geometric distribution for batch size. However, our model supports any batch size distribution and so it would be interesting to compare our model to measurements where the batch size distribution is based on statistics collected from real life I/O request stream data. Thirdly, caching is not yet supported in our model. Finally, we aim to combine our models for batch arrivals and variable-size arrivals into a single model to permit the representation of more realistic workloads with bursty, variably-sized I/O requests.

7. REFERENCES

[1] J. Abate and W. Whitt. The Fourier-series method for inverting transforms of probability distributions. *Queueing Systems Theory and Applications*, 10(1-2):5–88, 1992.

[2] M. L. Chaudhry and J. G. C. Templeton. *A First Course in Bulk Queues*. John Wiley & Sons, 1983.

[3] P. M. Chen, E. K. Lee, G. A. Gibson, R. H. Katz, and D. A. Patterson. RAID: High-Performance, Reliable Secondary Storage. *ACM Computing Surveys*, 26(2):145–185, June 1994.

[4] S. Chen and D. Towsley. The design and evaluation of RAID 5 and parity striping disk array architectures. *IEEE Transactions on Parallel and Distributed Systems*, 17(1-2):58–74, 1993.

[5] S. Chen and D. Towsley. A performance evaluation of RAID architectures. *IEEE Transactions on Computers*, 45(10):1116–1130, 1996.

[6] D. R. Cox. *Renewal Theory*. Chapman and Hall, 1962.

[7] H. A. David. *Order Statistics*. John Wiley and Sons, Inc, 1981.

[8] M. E. Gomez and V. Santonja. Characterizing temporal locality in I/O workload. In *Proc. International Symposium on Performance Evaluation of Computer and Telecommunication Systems (SPECTS '02)*, San Diego, CA, 2002.

[9] P. G. Harrison. Teaching M/G/1 theory with extension to priority queues. *IEE Proc. Computers and Digital Techniques*, 147(1):23–26, January 2000.

[10] P. G. Harrison and N. M. Patel. *Performance Modelling of Communication Networks and Computer Architectures*. Addison-Wesley, 1993.

[11] P. G. Harrison and S. Zertal. Queueing models of RAID systems with maxima of waiting times. *Performance Evaluation*, 64(7-8):664–689, August 2007.

[12] A. S. Lebrecht, N. J. Dingle, and W. J. Knottenbelt. Modelling and validation of response times in zoned RAID. In *Proc. 16th IEEE International Symposium on Modeling, Analysis, and Simulationof Computer and Telecommunication Systems (MASCOTS '08)*, September 2008.

[13] A. S. Lebrecht, N. J. Dingle, and W. J. Knottenbelt. A response time distribution model for zoned RAID. In *Proc. 15th International Conference on Analytical and Stochastic Modelling Techniques and Applications (ASMTA'08)*, June 2008.

[14] A. S. Lebrecht and W. J. Knottenbelt. Response time approximations in fork-join queues. In *Proc. 23rd UK Performance Engineering Workshop (UKPEW'07)*, July 2007.

[15] E. K. Lee. *Performance Modeling and Analysis of Disk Arrays*. PhD thesis, University of California at Berkeley, 1993.

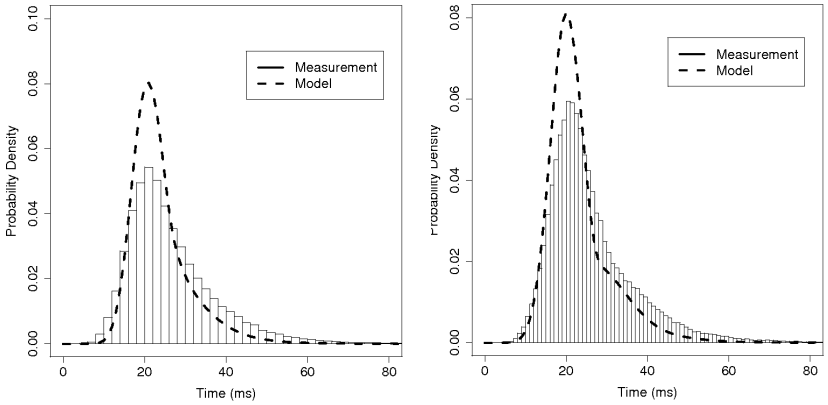
[16] D. Marquardt. An algorithm for least-squares estimation of nonlinear parameters. *SIAM Journal on Applied Mathematics*, 11:431–441, 1963.

[17] R. Nelson and A. N. Tantawi. Approximate analysis of fork/join synchronization in parallel queues. *IEEE Transactions on Computers*, 37(6):739–743, June 1988.

[18] R. Nelson, D. Towsley, and A. N. Tantawi. Performance analysis of parallel processing systems. *IEEE Trans. Softw. Eng.*, 14(4):532–540, 1988.

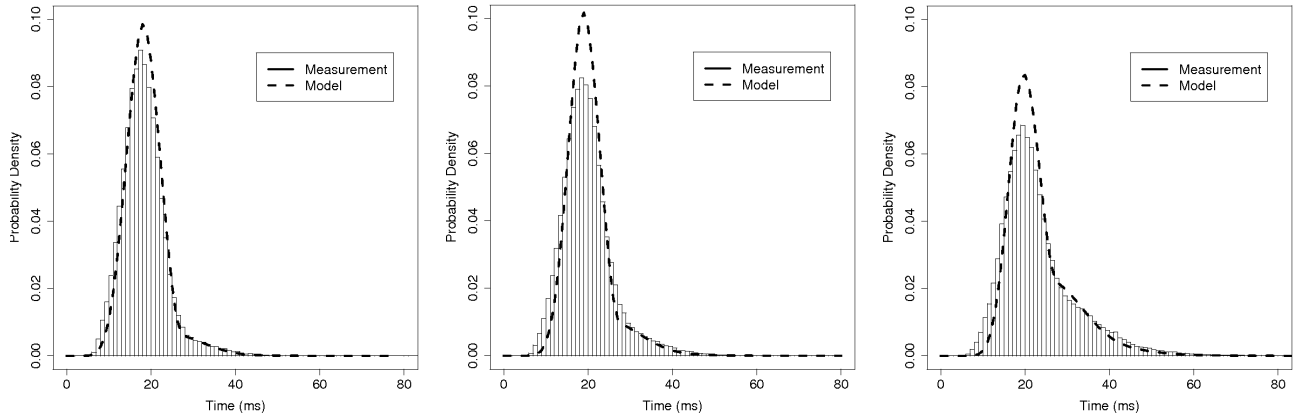
[19] A. Riska and E. Riedel. Disk drive level workload characterization. In *Proc. USENIX '06 Annual Technical Conference (ATEC '06)*, Boston, MA, 2006.

[20] C. Ruemmler and J. Wilkes. Unix disk access

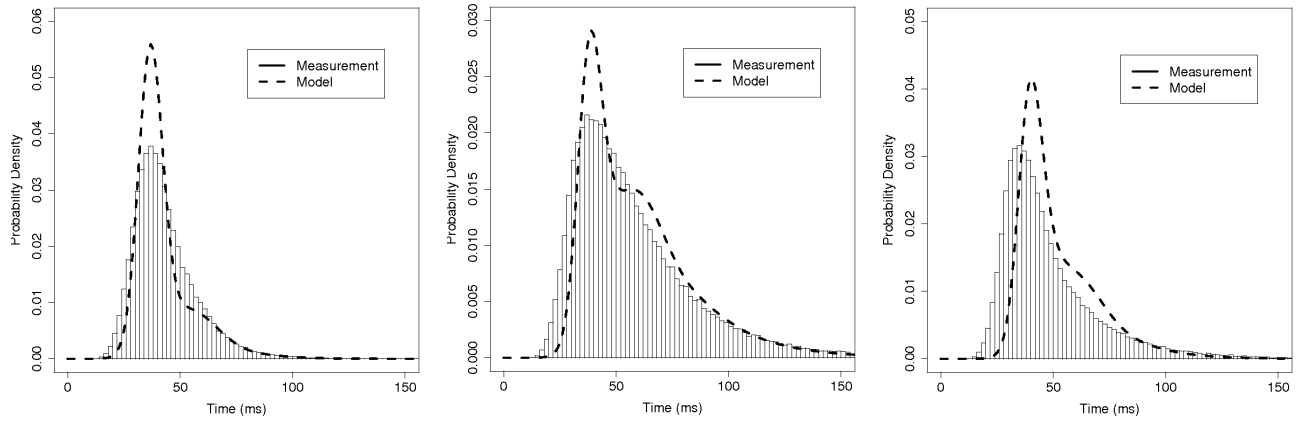


(a) mean job size = 4, 50% reads, $\lambda = 0.01$ (b) mean job size = 2, 25% reads, $\lambda = 0.02$

Figure 8: Response time pdf of model against measurement for requests on RAID 01 with different sized arrivals and a mix of reads and writes.



(a) mean job size = 2, reads, $\lambda = 0.01$ (b) mean job size = 3, reads, $\lambda = 0.01$ (c) mean job size = 4, reads, $\lambda = 0.02$



(d) mean job size = 2, writes, $\lambda = 0.01$ (e) mean job size = 3, writes, $\lambda = 0.02$ (f) mean job size = 5, writes, $\lambda = 0.01$

Figure 9: Response time pdf of model against measurement for requests on RAID 5 with different sized arrivals.

patterns. In *Proc. Usenix Winter Conference*, pages 405–420, San Diego, CA, 1993.

[21] J. Rydning and D. Reinsel. Worldwide hard disk drive 2009–2012 forecast: Navigating the transitions for enterprise applications. IDC Market Analysis, Document 216394, February 2009.

[22] Seagate. Barracuda ES Data Sheet, 2007. http://www.seagate.com/docs/pdf/datasheet/disc/ds_barracuda_es.pdf.

[23] E. Shriver, A. Merchant, and J. Wilkes. An analytic behavior model for disk drives with readahead caches and request reordering. In *Proc. 1998 ACM SIGMETRICS*, pages 182–191, Madison, WA, 1998.

[24] E. Varki. Mean value technique for closed fork-join networks. In *Proc. 1999 ACM SIGMETRICS*, pages 103–112, Atlanta, GA, 1999.

[25] E. Varki. Response time analysis of parallel computer and storage systems. *IEEE Transactions on Parallel and Distributed Systems*, 12(11):1146–1161, 2001.

[26] E. Varki, A. Merchant, J. Xu, and X. Qiu. Issues and challenges in the performance analysis of real disk arrays. *IEEE Transactions on Parallel and Distributed Systems*, 15(6):559–574, June 2004.

[27] M. Wang, A. Ailamaki, and C. Faloutsos. Capturing the spatio-temporal behavior of real traffic data. *Performance Evaluation*, 49(1-4):147–163, 2002.

[28] S. Wolfram. *The Mathematica Book*. Wolfram Media, 5th edition, 2003.

[29] S. Zertal and P. G. Harrison. Multi-RAID queueing model with zoned disks. In *Proc. High Performance Computing and Simulation Conference (HPCS'07)*, 2007.

APPENDIX

A. DISCRETE BACKWARD RECURRENCES

In this appendix we derive $G_Z(z)$ (used in Section 3). Let L be a discrete random variable representing the batch size for a randomly selected job with associated mass function $f_L(x) = \mathrm{P}(\text{A randomly selected job will have batch size } x)$. Note this is not the same as $f_B(x)$, the batch size pdf, as L is more likely to be a larger batch size, as there are more jobs in larger batches. L can be defined with a size-biased density:

$$f_L(x) = \frac{x f_B(x)}{E[B]}$$

The discrete random variable Z is the number of completed jobs in a batch during the service of a random job. Then,

$$P(Z = x \mid L = x_0) = \begin{cases} \frac{1}{x_0} & 0 \le x < x_0 \\ 0 & x \ge x_0 \end{cases}$$

By the law of total probability,

$$\begin{aligned} P(Z = x) &= \sum_{x_0=x+1}^{\infty} \frac{1}{x_0} f_L(x_0) \\ &= \sum_{x_0=x+1}^{\infty} \frac{f_B(x_0)}{E[B]} \qquad x = 0, 1, 2, \ldots \end{aligned}$$

Therefore the density and generating functions are,

$$\begin{aligned} f_Z(x) &= (1 - F_B(x))/E[B] \\ G_Z(z) &= \frac{1 - G_B(z)}{E[B](1 - z)} \end{aligned}$$