A Closed-Form Formula for Queueing Delays in Disk Arrays

Tao Yang [*] Shengbin Hu and Qing Yang [†]

1 Introduction

Disk arrays have become increasingly popular as a means of improving performance of secondary storage systems. In making design decisions, it is essential to understand the performances of different configurations as well as effects of system parameters on their performance. Analytical performance models are desirable in both configuring and designing disk arrays since they provide designers with a quick and efficient tool to evaluate and understand a system with a wide range of system and workload parameters.

Despite the importance of analytic performance models, very little has been reported in the literature on performance modeling of disk arrays. The main reason to this is that analytical models for disk arrays are difficult to formulate because of the presence of queueing and fork-join synchronization [1]. Most existing performance studies on disk arrays use time-consuming simulations with a few exceptions [2, 3, 4, 1, 5] which are summarized in [1]. Until now, however, no one has derived a queueing model that contains all of the following important and realistic properties:

1. A logical disk request, often called *array request* [1], may fork into several disk requests that are served in parallel by a group of disks;
2. Array request size, i.e. the number of disks involved in serving a logical disk request, can take different values with respect to the total number of disks in the disk array system depending on the parity scheme, parity group size, degree of declustering [6], and request type. It can not be guaranteed that all logical disk requests require either only one disk or all the disks in the system;
3. The disk requests belonging to an array request may finish their data accesses at different times due to the randomness of head positions and queueing effects in disks. This is true for all RAID systems which operate asynchronously;
4. Depending on the I/O workloads, it can not be guaranteed that there is always a new array request being issued immediately after a disk finishes its data access for a request.

[*]Partially supported by the Natural Sciences and Engineering Research Council of Canada under grant No. OGP004-6415, The author is with Dept. of Industrial Engineering, Technical University of Nova Scotia, Halifax, Nova Scotia, Canada B3J 2X4

[†]Partially supported by National Science Foundation under grant MIP-9208041. The authors are with Dept. of Electrical and Computer Engineering, The University of Rhode Island, Kingston,RI, USA 02881 e-mail: qyang@ele.uri.edu

We present here an analytical model that contains all these four properties for disk arrays. More importantly, our model gives a closed-form solution to the expected response time of an array request with a similar approximation made by Kim and Tantawi in [4] (Eq. (7)) that was validated through real measurements. The model is based on a queue network consisting of processors and a disk array system queue. Customers or jobs that circulate in the queue network are *array requests* each of which consists of a fixed number (array request size) of disk requests. When an array request is generated from a process, the disk requests belonging to the array request are placed in corresponding queues of involved disks. Each disk serves disk requests in the queue on first-in-first-out (FIFO) basis. An array request is complete only when all the disk requests belonging to that array request have finished their disk accesses. As a result, if one considers each array request as a queue customer and looks at the disk array system queue as a whole, the queue service discipline is not necessarily first-in-first-out any more because some disks may be more heavily loaded than others. This phenomenon makes the analytical model nontrivial. We have devised an approximation technique to handle this kind of irregular queue service discipline. The analytical model is validated through extensive simulation experiments to show they are in a very good agreement.

2 Queueing Model

We model the disk array system as a two-node closed queueing network as shown in Figure 1. Each *job* in the queueing system represents a logical I/O request or *array request* issued by a processor. The first node is a conventional infinite-server queue. The event that job i is in the first node implies that the corresponding process i is busy doing computation. A process makes an I/O request after a certain amount of time when a file needs to be loaded, a page fault occurs or alike. This amount of computation time varies depending upon the applications. It is assumed that the interval time between two consecutive I/O requests is an exponentially distributed random variable with mean Z.

The second node consists of D parallel single-server queues, each of which represents an individual disk and the associated queue in the disk array of D disks. Upon arrival at the second node, a job will split into n *subjobs* each of which represents an individual *disk request*. The n subjobs will join n consecutive disk queues starting from one of D disks equally likely. Subjobs arriving at a disk queue are served on the first-in-first-out basis. A job remains in the second node until all its subjobs are served. Although all the subjobs in a disk queue are processed in the order of arrival, the same does not

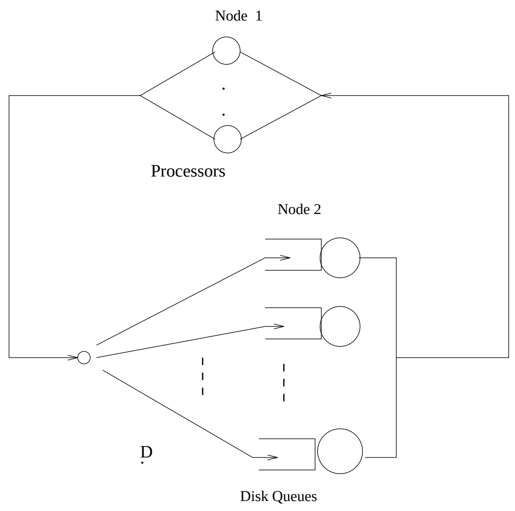

Figure 1: The queueing model.

necessarily apply to jobs as explained in the last section.

All subjobs are identical in the sense that they request exactly same amount of data such as a byte, a block or a sector depending upon the degree of interleaving. However, the service times of subjobs of a job vary depending upon disk head positions and requested-data locations. The service time of a subjob or a disk request consists of three parts: *seek time* for the disk head to move to the disk track containing the requested data, *rotation latency* for the desired sector to rotate under the head position before it can be read or written, and *transfer time* for the requested data to be physically transferred from the disk to a buffer from where the CPU will read complete data after all subjobs are done. Previous research and experiments [1, 4] have shown that the *seek time* is in the following form:

$$C_a\sqrt{X} + C_bX + C_c; \quad for \quad X \geq 0,$$

where C_a, C_b, and C_c, are constant chosen to satisfy some mechanical and electrical constraints [1], and X is a random variable. Most previous analytical models for disk arrays have assumed that X is uniformly or exponentially distributed [4]. In reality, X is the difference in terms of track numbers between the current and desired positions of the disk head. Both the current head position and the position of the requested data track are uniformly distributed between 0 and T, where T is the maximum number of tracks on one side of a disk. To make our model more realistic, we consider X to be the absolute difference between two uniform random numbers. Taking X to be the absolute difference of two uniform random numbers makes significant difference in result from assuming X to be a uniform random number. The former gives $T/3$ as the mean of X while the latter gives $T/2$ as the mean of X [7].

The *rotation latency* generally depends on the speed of disk rotations. For today's disks, a full revolution takes a fixed amount of time that is usually about 16 ms. It is reasonable to assume that the rotation latency to be a uniform random variable between 0 and L, where L is the length, in time, of one revolution time. Let T_R represent the random variable for rotation latency. Clearly, T_R is independent of seek time and is uniformly distributed between 0 and L. The *transfer time* is generally deterministic once the degree of interleaving is determined. Let T_t represent the data transfer time. We have the service time of a disk queue in the second node of the queue network

$$S = C_a\sqrt{X} + C_bX + C_c + T_R + T_t; \quad for \quad X \geq 0, \quad (1)$$

where C_a, C_b, C_c, and T_t are constant and, X and T_R are mutually independent random variables.

Consider an arbitrary queue in the second node. Whenever there is an arrival at the second node, this queue will receive a subjob with probability n/D since the starting block of an array request is uniformly distributed over the D disks. Assuming that each processor performs context switching after issuing an array request, the arrival process at each individual queue is therefore a Poisson process with rate λ which is given by

$$\lambda = \frac{n}{D}\frac{P}{Z}.$$

Hence the second node can be considered as consisting of D M/G/1 queues each of which has a service time S given in Equation (1).

Let $B(x) = Pr\{S \leq x\}$ be the cumulative distribution function of S. Let Y_i be the waiting time (including its service time) of an arbitrary subjob that joins the i^{th} queue in the second node and let $G_i(y) = Pr\{Y_i \leq y\}$ be its cumulative distribution function. It can be shown that all Y_i's have the same distribution. Let W be the total time spent in the second node by an arbitrary job and $H(x) = Pr\{W \leq x\}$ be its cumulative distribution function. Since a job remains in the second node until the last subjob completes its disk access, W is given by

$$W = max(Y_1, Y_2, \cdots, Y_n). \quad (2)$$

We notice that the random variables Y_1, Y_2, $\cdots$, and Y_D are not necessarily independent. However, since their correlation is quite weak from our experience in modeling similar systems [8] and the simulation experiments presented later in this paper, the approximation assumption that they are independent is expected to cause insignificant error for many cases of practical interest. Under the independence assumption, we have

$$H(x) = G^n(x), \quad (3)$$

and the expected waiting time $E(W)$ is given by

$$E(W) = \int_0^\infty (1 - G^n(x))dx. \quad (4)$$

In order to derive $E(W)$, let us first consider Y_i, the waiting time of each individual queue. Let $\mu_Y (= E(Y))$ and σ_Y^2 be the mean and variance of Y, respectively. From queueing theory, we obtain

$$\mu_Y = E(Y) = E(S) + \frac{\lambda E(S^2)}{2(1-\rho)}, \quad (5)$$

and

$$\sigma_Y^2 = E(S^2) - E^2(S) + \frac{\lambda^2 E^2(S^2)}{4(1-\rho)^2} + \frac{\lambda E(S^3)}{3(1-\rho)}, \quad (6)$$

where $\rho = \lambda E(S)$.

To find the moments $E(S)$, $E(S^2)$ and $E(S^3)$, we notice that S is the sum of two independent random variables S_1 and S_2, where $S_1 = C_a\sqrt{X} + C_b X$ and $S_2 = C_c + T_R + T_t$. Therefore, we have

$$\begin{array}{ll} E(S) & = E(S_1) + E(S_2), \\ E(S^2) & = E(S_1^2) + 2E(S_1)E(S_2) + E(S_2^2), \\ E(S^3) & = E(S_1^3) + 3E(S_1^2)E(S_2) \\ & +3E(S_1)E(S_2^2) + E(S_2^3). \end{array} \tag{7}$$

Since S_2 is uniformly distributed between $\alpha = C_c + T_t$ and $\beta = C_c + T_t + L$, we have

$$\begin{array}{ll} E(S_2) & = (\alpha+\beta)/2 = C_c + T_t + L/2, \\ E(S_2^2) & = (\beta^3 - \alpha^3)/(3L), \\ and & \\ E(S_2^3) & = (\beta^4 - \alpha^4)/(4L), \end{array} \tag{8}$$

Let $B_1(x)$ be the cumulative distribution function of the random variable S_1. Then we have

$$Pr\{S_1 \le x\} = Pr\{C_a\sqrt{X} + C_b X \le x\}$$

$$= Pr\left\{\sqrt{X} \le \frac{\sqrt{C_a^2 + 4C_b x} - C_a}{2C_b}\right\}. \tag{9}$$

Since $Pr\{X \le x\} = (2xT - x^2)/T^2, 0 \le x \le T$, which is the distribution of the absolute difference between two uniform random variables, we obtain

$$B_1(x) = \begin{cases} \frac{f(x)(2T - f(x))}{T^2}, & if\ x \le \gamma, \\ 1, & otherwise, \end{cases} \tag{10}$$

where $\gamma = \sqrt{T}C_a + TC_b$ and $f(x) = \left(\frac{\sqrt{C_a^2+4C_b x} - C_a}{2C_b}\right)^2$. Let $b_1(x) = dB_1(x)/dx$ for $0 \le x \le \gamma$. Then, we have for $0 \le x \le \gamma$,

$$b_1(x) = \frac{2f'(x)}{T} - \frac{2f(x)f'(x)}{T^2}$$

$$= \frac{1}{T^2 C_b^3}\left[2TC_b^2 - 2C_a^2 - 2C_b x + \right.$$

$$\left. + C_a\sqrt{C_a^2 + 4C_b x} - \frac{C_a(2TC_b^2 - C_a^2 - 2C_b x)}{\sqrt{C_a^2 + 4C_b x}}\right]. \tag{11}$$

Hence we obtain

$$E(S_1) = \int_0^\gamma x b_1(x)dx, \tag{12}$$

$$E(S_1^2) = \int_0^\gamma x^2 b_1(x)dx, \tag{13}$$

and

$$E(S_1^3) = \int_0^\gamma x^3 b_1(x)dx. \tag{14}$$

Because of the nice form of Equation (11), we can derive closed form expressions for $E(S_1)$, $E(S_1^2)$ and $E(S_1^3)$ from the integrations in Equations (12), (13), and (14). Substituting the expressions for $E(S_1)$, $E(S_1^2)$ and $E(S_1^3)$ into Equation (7), we obtain the first, second and third moments of S. Using Equations (5) and (6), we get closed-form expressions for μ_Y and σ_Y. According to (4) and the approximation in [4], we have

$$E(W) = \mu_Y + \sigma_Y\sqrt{2log(n)}. \tag{15}$$

And a tight upper bound on $E(W)$ is given by

$$E(W) \le \mu_Y + \sigma_Y\frac{n-1}{\sqrt{2n-1}}. \tag{16}$$

3 Experimental Validation

In order to verify the correctness of our analytical model developed in the last section, we have developed a simulator. There are two main approximations in our analysis: independence assumption to derive Equation (3) and the approximation of $E(max(Y_i))$ in Equation (15). Our main purpose here is to analyze the effects of these two approximations. We run our simulator in two different modes. In the first mode, the starting disk of an array request is selected from the D disks equally likely. Once the starting disk is determined, n disk requests are then appended to the subsequent disk queues. As a result, dependency between disks exists. With this mode, we check the errors of the analytical model caused by the independence assumption. The second mode of the simulator assumes independence between disk queues, i.e. n disks are selected randomly to serve an array request upon arrival. With this mode, we analyze the sole effect of approximation in Equation (15). All simulations have been run for sufficiently long time so that the 90% confidence intervals are small enough to be within 1%.

We validate our analysis by considering all possible workload parameters and the validation is done in terms of response times. Recall that the load to the queueing system is mainly determined by the arrival rate λ which is a function of D, n, Z, and P. We observed through our experiments that the effects of changing $1/Z$ and P are identical, therefore we consider all possible values of D, n, and Z. By all possible values, we mean the values of D, n, and Z that make ρ range from very small to close to 1. Other system parameters are fixed at the commonly used values in the literature : $C_a = 0.4623$, $C_b = 0.0092$, $C_c = 2$, $T = 949$, subblock size is 4 Kbytes and the average transfer time is 0.6023 ms/Kbyte.

Figure 2 shows the comparison between our analysis and the simulation. Response time of array requests are drawn as a function of ρ which is the utilization of each disk queue. Different ρ's in Figure 2 are obtained by varying D between 30 and 100. It can be seen from this figure that our analytical model matches well with both simulation results. It is clear from this figure that the assumption that disk queues are independent brings little error. Even at very high load, i.e. ρ approaching 1, the results with independence assumption are very close to that without independence assumption. Our analysis

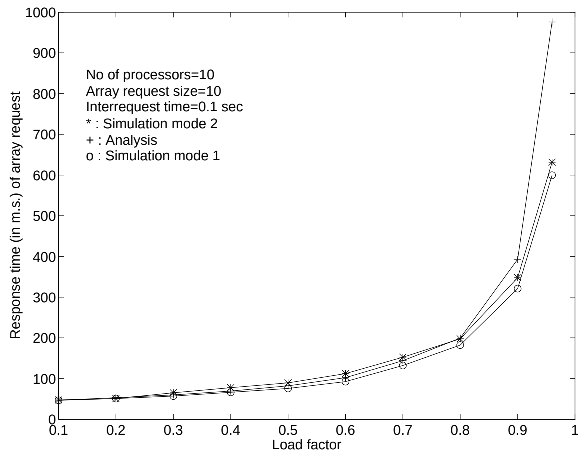

Figure 2: Response time vs ρ by varying D.

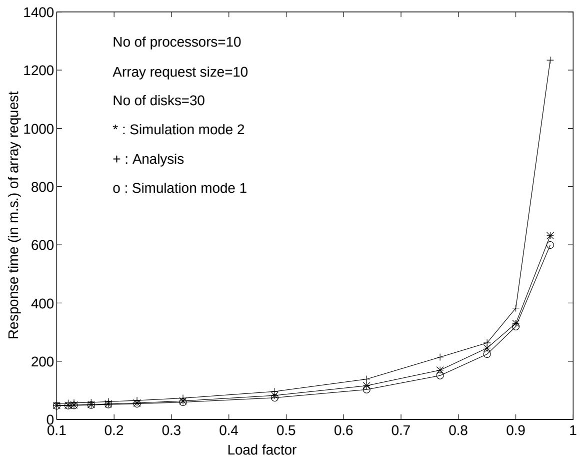

Figure 4: Response time vs ρ by varying Z.

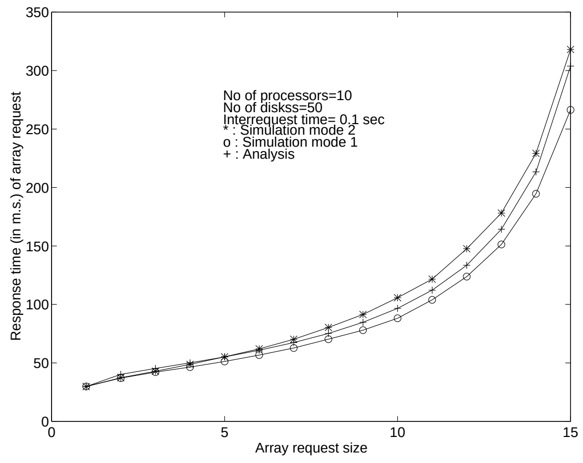

Figure 3: Response time vs array sizes n.

slightly over estimates the queueing delay at very high load as shown in this figure. Such over estimation can be mainly attributed to the independence assumption and the approximation of Equation (15).

In Figure 3, we plotted the analytical and simulation results by varying array request size n. Again, the analytical results are very close to those of simulations and have the exactly the same shape as that of simulations. As the array request size increases, we observed slight increase in the difference between the analysis and simulations. This increase is under our expectation because the larger the n is, the stronger the dependence between disk queues will be. As a result, the model over estimates the delay. But, we notice that the error caused by the second approximation (15) does not increase with the increase of n. In general, the two results match well. Similar agreements are observed by varying interrequest time Z as shown in Figure 4.

4 Conclusions

In this paper, we have presented an approximate analytical performance model for disk array systems. We consider the request forking, queueing, and nonFIFO service for array requests as is the case in reality. Based on these assumptions, we have derived a closed-form solution for the expected queueing delay of an array request in the disk array system. Simulation experiments have been carried out to validate our analysis. It has been shown that the newly developed analytical model behaves well for all practical workloads. The closed-form formula can be used easily by disk array designers to parameterize the system with no cost.

References

[1] E. K. Lee and R. H. Katz, "An analytic performance model of disk arrays," in *Proc. of ACM SIGMETRICS*, 1993.

[2] D. Bitton and J. Gray, "Disk shadowing," in *Proc. of Very Large Data Bases*, pp. 331–338, 1988.

[3] M. Y. Kim, "Synchronized disk interleaving," *IEEE Trans. on Computer*, vol. C-35, pp. 978–988, Nov. 1986.

[4] M. Y. Kim and N. Tantawi, "Asynchronous disk interleaving: approximate access delays," *IEEE Trans. on Computer*, pp. 801–810, Feb. 1989.

[5] K. Salem and H. Garcia-Molina, "Disk striping," *IEEE Data Engineering*, pp. 336–342, Feb. 1986.

[6] A. L. N. Reddy and P. Banerjee, "An evaluation of multiple-disk I/O systems," *IEEE Trans. on Comput.*, vol. 38, pp. 1680–1690, Dec. 1989.

[7] K. S. Trivedi, "Probability and statistics with reliability, queueing, and computer science applications," *Prentice Hall*, 1982.

[8] Q. Yang, L. Bhuyan, and B.-C. Liu, "Analysis and comparison of cache coherence protocols for a packet-switched multiprocessor," *IEEE Trans. on Comput.*, pp. 1143–1153, August 1989.