Models of DASD Subsystems with Multiple Access Paths: A Throughput-Driven Approach

ALEXANDRE BRANDWAJN

Abstract—**A throughput-driven approach to the performance analysis of direct access storage devices (DASD's), i.e., disks, is presented for a model of a block-multiplexor channel system with multiple transfer paths. The approach relies on a step-wise analysis of shared components within the data transfer path: head of string, control unit, channel, referred to as blocking points. In order to derive the service time of an I/O request, each blocking point is analyzed separately, using essentially a classical loss-system model. The advantages of the method are its relative simplicity and applicability to a large number of configurations. Transient and boundary effects, including the miss probability on second and subsequent reconnection attempts, and the device reconnection window (lead time), are also discussed. The inclusion of the approach in a larger system model, as well as its relationship with previous work on DASD analysis is considered. Numerical results illustrate the accuracy of the method.**

Index Terms—**Analytical models, block-multiplexor channels, disk subsystems, dual ports, missed reconnections, multiple transfer paths, reconnection window, shared control units, string switching, system model.**

I. Introduction

In a previous paper [3], we have presented one approach to the modeling of direct access storage device (DASD) subsystems. This approach, designed for use with the equivalence method of analysis of queueing models, results in a solution of a set of submodels, and computes the I/O request throughput as a primary performance measure. By design, it is well suited for use in a larger system model. A drawback of this approach, however, is that the number of times the submodels have to be evaluated grows with the number of users in the system. Another drawback is that extensions to other configurations may be difficult.

In this paper we propose a different approach, with different advantages and drawbacks. The approach takes the I/O request throughput as a starting point, and derives the DASD (disk) basic service time pertaining to the given throughput. By definition, the basic I/O service time is taken here to denote the average time it takes to service an I/O request from the moment it has been successfully issued. The total I/O service time is the the average time it takes to service an I/O request from the moment the CPU first attempts to issue it. The difference between the total and the basic time is viewed as a queueing delay. Having obtained the basic service time we estimate to total I/O service time by viewing the devices as separate single-server queues.

With block-multiplexor channels—the only type considered herein—three elements in the data path between the CPU and the DASD may be shared by spindles. Devices are configured into strings where all the disks on a string may be simultaneously moving arms to the appropriate track (seek) or waiting for the read/write heads to reach the required angular position (latency). However, in most systems, only one disk may be transferring using the necessary hardware within the head of string. Several such strings may be connected to a single control unit, and several control units may share a single channel. Also, several channels may be linked to the same control unit, resulting in a more or less complex configuration. All three elements, viz. head of string, control unit and channel, are potential blocking points in that they must be available at the moment a device is about to reach the correct angular position in order for the disk to reconnect successfully and start transferring. Should any of the elements be busy with another device, a missed reconnection occurs, and the I/O request experiences the heavy penalty of a full rotation period before another reconnection attempt is possible.

In this paper, we neglect the dependence of the seek time on the load, and concentrate on missed reconnections as the principal load-dependent component of the I/O request basic service time. Each blocking point is studied separately, so that the approach taken copes easily with multiple blocking points as encountered in more complex configurations. The analysis is introduced for a model with multiple data transfer paths in control units and strings of disks. This provides a framework for modelling both existing and possibly forthcoming products.

In the next section, this approach is presented for a simplified case with balanced I/O rates and utilizations. Transient and boundary effects, such as nongeometrical distribution of the number of missed reconnections, or effect of the reconnection window (disk lead time), are also discussed in this section. Application of the approach to present-day configurations with shared channels, shared strings (string switching), and disks with dual ports (cross-call) is considered in Section III. The relationship of the method with other approaches to DASD analysis [2], [17], [10] is also discussed. Section IV briefly considers the embedding of the solution of a DASD subsystem model in that of a larger system model. Using a simple example of a DASD subsystem, a comparison of the results of the method of this paper and of [3] is presented. Finally, Section V summarizes the advantages and the drawbacks of the method.

Manuscript received March 2, 1981; revised December 21, 1981, and June 30, 1982.

The author is with the Amdahl Corporation, 1250 East Arques Avenue, Sunnyvale, CA 94086.

0018-9340/83/0500-0451$01.00

II. Multiple Access Paths

A. Configuration Considered and Assumptions

To introduce our approach, we consider a hypothetical DASD subsystem in which every channel is connected to only one control unit, and a control unit has N_c channels connected to it. For our purpose in this section, it is irrelevant whether these channels belong to a single CPU or different CPU's. We assume that a control unit has N_r independent nondedicated data transfer paths, henceforth referred to as *routes*. We also assume that there are N_s strings of DASD's connected to the storage director, and that within every string there are N_p independent nondedicated data transfer paths shared by the disks on the string. These access paths within a string will be henceforth referred to as *paths*. For simplicity, we assume that there are N_d devices in every string attached to the control unit.

A sample configuration of a control unit is represented in Fig. 1. The control unit shown has four channels connected to it. It possesses two routes, and the single string of DASD's attached to it has two paths. Thus, a maximum of two devices may be simultaneously engaged in transfer in this example.

It is now our goal to analyze the hypothetical DASD subsystem so as to assess the impact of multiple routes and multiple paths on the I/O service time. We view the total I/O service time of a DASD as composed of the elements represented below.

Queueing	Seek	Latency	Missed Reconnection Delay	Transfer
			Basic I/O Service Time	

The seek time corresponds to arm movement until the correct track is reached. The latency is the time for the device to reach the required angular position in its rotation. The missed reconnection delay is that part of the basic I/O service time due to the disk finding the I/O path busy when the appropriated sector is about to be reached.

Referring to the given control unit, we denote by t_s, t_l, t_m, and t_t the average seek, latency, missed reconnection delay, and transfer time, respectively. The device rotation period is denoted by t_r.

We proceed to evaluate the basic I/O service time under the following assumptions.

1) All I/O requests are statistically identical, and are evenly distributed over channels and devices. This assumption is for exposition purposes only, and is relaxed in [4].

2) The dependence of the seek time on the load is not represented. A separate model of arm movement would have to be added if such a contention were a dominant factor determining the performance of the DASD subsystem.

3) Elongations in the basic I/O service time due to the I/O path being busy when a CPU attempts to issue the I/O request (redriven I/O's) or after the completion of a seek when the target angular position has to be forwarded to the device, are neglected.

4) The operation of count-key-data devices is implicitly represented by the transfer time. The latter, for count-key-data devices, is taken to include the search operation during which a key match is sought.

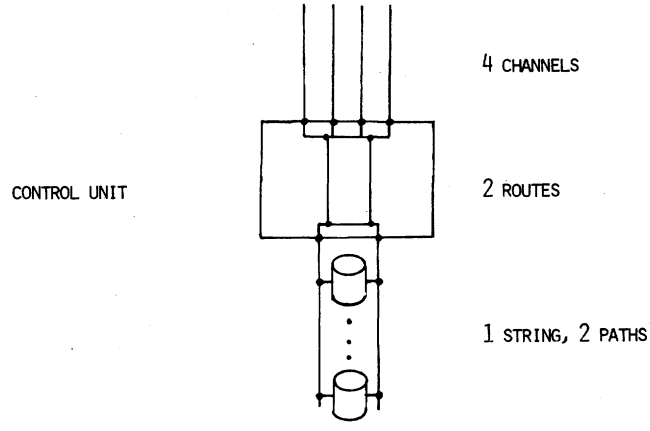


Fig. 1. A sample configuration with multiple paths and routes.

Additional assumptions, and departures from the above ones, are explicitly indicated where appropriate.

B. Analysis of Blocking Points

In this section we analyze the state of the data transfer path encountered by a device reconnection request. The following principal notations are used in this section.

N_d: number of DASD's (actuators) in a string.

N_s: number of strings connected to the control unit.

N_p^*: maximum number of paths which may be simultaneously active within a string.

N_r^*: maximum number of routes which may be simultaneously active at the control unit.

N_h^*: maximum total number of paths requesting reconnection to the control unit.

θ: I/O throughput of the control unit.

θ_s: I/O throughput of a string of DASD's.

λ_p: $1/\lambda_p$ is the average time between consecutive reconnection requests from a DASD not engaged in transfer.

λ_r: $1/\lambda_r$ is the average time between consecutive reconnection requests from an idle path in a string.

μ_p: rate of transfer completions of an active string path.

μ_r: rate of transfer completions of an active control unit route.

a_j: probability that an arriving device reconnection request finds the appropriate I/O path available given that j paths are found active at the device's string.

b_k: probability that the channel requested by the control unit is found available given that k routes are already busy at the control unit.

$p_s(j)$: stationary probability that j paths are busy at a string, $j = 0, \cdots, N_p^*$.

$P_s(j)$: steady-state probability that a DASD requesting reconnection finds j paths busy within its string.

$p_r(k)$: stationary probability that a k routes are busy at the control unit, $k = 0, \cdots, N_r^*$.

$P_r(k)$: probability that a reconnection request from a string finds k routes busy at the control unit.

Consider a control unit of the hypothetical DASD subsys-

tem, and denote by θ its I/O throughput (i.e., the average number of I/O requests completed per unit time). θ is assumed to be known and is the basis for the following analysis. In order to evaluate the probability of a missed reconnection, we consider blocking points separately, starting from string paths, and moving towards the CPU. We thus most naturally follow a reconnection request as it makes its way through the I/O path. Each blocking point is viewed as a simple loss system [6]. The dependence between blocking points is taken into account via appropriate activation rates for the resources of the blocking point.

Denote also by N_r^* the maximum number of routes within the control unit that may be simultaneously engaged in transfer. We have

$$N_r^* = \min(N_r, N_c, N_s \cdot \min(N_p, N_d)).$$

Let N_p^* be the maximum number of paths within a string of DASD's which may be simultaneously transferring. We have

$$N_p^* = \min(N_r^*, N_p, N_d).$$

Having thus accounted for various awkward cases where more hardware is provided than can be actually utilized, we proceed to the analysis of the missed reconnection delay. We start with the paths at the head of string, and we denote by λ_p the rate of path reconnection requests issued by a DASD on the string, and by μ_p the rate of completions of transfers by a path. Neglecting overheads, we have $\mu_p \simeq 1/t_t$. We make the additional assumption that the path reconnection requests from a disk constitute a Poisson process. With this assumption, the equilibrium probability that j paths are simultaneously busy at the string, denoted by $p_s(j)$, is given by (cf. [6])

$$p_s(j) = \frac{1}{G_s} \prod_{i=1}^{j} \lambda_p (N_d - i + 1) a_{i-1}/(i\mu_p), \qquad (2.1)$$

$$j = 0, \cdots, N_p^*,$$

where G_s is a normalization constant, and a_j is, by definition, the probability that all other necessary elements of the I/O path (route and appropriate channel) are found available by a device requesting reconnection given that j paths are active at that string.

Let $P_s(j)$ be the steady-state probability that a DASD requesting reconnection finds j paths busy within its string. The number of reconnection requests which arrive per unit time to find such a situation is simply $(N_d - j)\lambda_p p_s(j)$. Hence,

$$P_s(j) = \frac{(N_d - j)p_s(j)}{\sum_{k=0}^{N_p^*} (N_d - k)p_s(k)}, \qquad j = 0, \cdots, N_p^*. \qquad (2.2)$$

Note that (2.1) and (2.2) are valid regardless of the distribution of path busy times (see, e.g., [6]). Note also that, in order for these expressions to be useful, we must know the probability a_j and the request rate for a single device λ_p. Suppose for the moment that a_j is known. λ_p can be determined as follows. Let θ_s be the I/O throughput of the given string of DASD's. With balanced load distribution, as assumed herein, $\theta_s = \theta/N_s$. We have under equilibrium

$$\sum_{j=1}^{N_p^*} p_s(j) j \mu_p = \theta_s. \qquad (2.3)$$

Using (2.1) in (2.3) we obtain the following polynomial equation for the rate of reconnection requests λ_p

$$\sum_{j=1}^{N_p^*} \rho_p^j \left[\prod_{i=1}^{j} (N_d - i + 1) a_{i-1}/i \right] (j - \rho_s) - \rho_s = 0 \qquad (2.4)$$

where

$$\rho_p = \lambda_p/\mu_p \quad \text{and} \quad \rho_s = \theta_s/\mu_p.$$

Formula (2.4) is a low degree polynomial equation so that its solution for ρ_p (and hence λ_p) may be obtined by any standard method. We note that the rate of reconnection requests from a device cannot be lower than its rate of successful reconnections

$$\rho_p \geq \rho_s/N_d.$$

Therefore, a simple bisection over the interval, say, $(\rho_s/N_d, 10\rho_s/N_d)$ has been found a satisfactory approach to the solution of (2.4).

a_i, the probability of finding a route and the appropriate channel available given that j devices are transferring at the string, is obtained by analyzing the two blocking points routes and channels. Let $p_r(k)$ be the steady-state probability that k routes are active, $k = 0, \cdots, N_r^*$. We now view the paths (at the heads of string) as sources of requests for routes. The number of such sources is denoted by N_h and is given by

$$N_h = N_s \cdot \min(N_p, N_d).$$

Denote by μ_r the rate of transfer completions for an active route. Neglecting overheads, we have $\mu_r = 1/t_t$. We make the assumption that the requests issued by each of the N_h sources constitute a Poisson process, and we denote by λ_r the corresponding rate. With this assumption $p_r(k)$ is simply given by

$$p_r(k) = \frac{1}{G_r} \prod_{i=1}^{k} \lambda_r (N_h - i + 1) b_{i-1}/(i\mu_r), \qquad (2.5)$$

$$k = 0, \cdots, N_r^*$$

where G_r is a normalization constant, and b_k is the probability that the appropriate channel is found available by a route request which is issued when k routes are already busy. Let $P_r(k)$ be the probability that a route request finds k routes engaged in transfer. The number of route requests arriving per unit time to find k routes busy is $(N_h - k)\lambda_r p_r(k)$. Hence,

$$P_r(k) = \frac{(N_h - k)p_r(k)}{\sum_{j=0}^{N_r^*} (N_h - j)p_r(j)}, \qquad k = 0, \cdots, N_r^*. \qquad (2.6)$$

As before, in order for the above formula to be usable, we have to know the probabilities b_k and the request rate λ_r. Suppose b_k is known. λ_r may then be determined as follows. The I/O throughput handled by the set of routes is, of course, the total throughput of the control unit θ. Thus

$$\sum_{k=1}^{N_r^*} p_r(k) k \mu_r = \theta. \tag{2.7}$$

Substituting (2.5) in (2.6) we get the following polynomial equation for the rate of route requests λ_r

$$\sum_{k=1}^{N_r^*} \rho_r^k \left[\prod_{i=1}^{k} (N_h - i + 1) b_{i-1}/i \right] (k - \rho) - \rho = 0 \tag{2.8}$$

where $\rho_r = \lambda_r/\mu_r$, and $\rho = \theta/\mu_r$. Since we must have

$$\rho \geq \rho/N_h,$$

formula (2.8) may be easily solved for ρ_r by simple bisection over the interval, say, $(\rho/N_h, 10\rho/N_h)$.

Assuming requests for channels from routes arrive at random with respect to the status of the appropriate channel, we readily obtain

$$b_k = 1 - k/N_c. \tag{2.9}$$

This completes the analysis of the routes and channels, and allows us to determine a_j, the probability of finding a route and the required channel available given that j paths are busy at a string. Denote by $M(j)$ the maximum number of routes which may be simultaneously engaged in transfer when j paths are busy at the tagged string. We have

$$M(j) = \min (N_r^*, (N_s - 1) \cdot N_p + j).$$

Let $N(j)$ be the maximum number of routes which may be found active while still leaving at least one useful route available with j paths busy at the string. $N(j)$ is given by

$$N(j) = \min (N_r^* - 1, (N_s - 1) \cdot N_p + j).$$

a_j is then simply

$$a_j = \sum_{k=j}^{N(j)} P_r(k) b_k \Big/ \sum_{k=j}^{M(j)} P_r(k). \tag{2.10}$$

Having determined all the elements necessary to compute the state probabilities of the blocking points as seen by arriving reconnection requests, we are now ready to evaluate the probability of a missed DASD reconnection. The latter occurs if all the paths at the string are found busy, or if, at least one path being available, the remaining part of the I/O path (routes, appropriate channel) is found busy. Let p_{m_1} be the corresponding probability. We have

$$p_{m_1} = \dot{P}_s(N_p^*) + \sum_{j=0}^{N_p^*-1} P_s(j)(1 - a_j),$$

which may be rewritten as

$$p_{m_1} = 1 - \sum_{j=0}^{N_p^*-1} P_s(j) a_j. \tag{2.11}$$

Hence, denoting the average number of missed reconnections per successful I/O request by n_m, we obtain, as a first approximation,

$$n_m \simeq n_{m_1} \triangleq p_{m_1}/(1 - p_{m_1}), \tag{2.12}$$

and the missed reconnection delay is

$$t_m = t_r n_m.$$

Denoting by W_b the basic I/O service time, we have

$$W_b = t_s + t_l + t_m + t_t. \tag{2.13}$$

Note that in our analysis string paths and control unit routes are taken to be nondedicated in the sense that any path and route available. Channels, however, are assumed to be conventional, and an I/O initiated on a given channel is bound to that channel until completion. If this were not the case, i.e., if the channels were "floating," we would simply have $b_k = 1$, $k = 0, \cdots, N_r^* - 1$.

We now recap the solution procedure, moving backwards as necessary in actual computation. First, the probabilities b_k are computed. Then, (2.8) is solved, and $p_r(k)$ and $P_r(k)$ are computed from (2.5) and (2.6), respectively. Next, a_j is obtained from (2.10) and used in the solution of (2.4) to compute $p_s(j)$ and $P_s(j)$ from (2.1) and (2.2), respectively. Finally, (2.11) is used to evaluate p_{m_1}. Note that from (2.11) one can easily compute the probability that a missed reconnection is caused by a given blocking point.

We have not investigated theoretically the uniqueness of the solution of the polynomial equations (2.4) and (2.8). On intuitive grounds, one would expect the I/O throughput to be a nondecreasing function of the rate of reconnection requests, and those two equations to possess a unique solution in the feasible region. In practice, in the many cases considered, the bisection has never failed to produce a solution.

C. Transient and Boundary Effects

In this section we take a brief look at a few phenomena which can affect the missed reconnection probability. The first such an effect is the nongeometrical distribution of the number of missed reconnections per transfer.

1) Correction for Missed Reconnection Probability: Our analysis of blocking points assumes that reconnection requests arrive with individual intensity independent of the status of the blocking point. This assumption of randomness is usually justified for a first reconnection attempt after initial positioning (cf. Section II-C2). It is recognized in the literature (e.g., [2], [17]) that subsequent reconnection attempts experience higher miss probabilities. Various explanations, such as disks "getting out of sync" or requests experiencing higher than average congestion spike, have been advanced for this phenomenon, and mostly empirical corrections for the miss probability have been proposed [2], [17].

We think that it is possible to view this phenomenon as a transient probability effect. For simplicity, consider a system with a single blocking point, such as one string of disks connected to a single channel. We shall refer to this blocking point as the unit. Consider also a reconnection request having experienced a first miss, i.e., having found the unit busy when the appropriate sector was reached. The next reconnection request from this device will come t_r time units later, after a full revolution of the disk. What this next request will "see" as unit busy probability is the transient probability of the unit being busy t_r time units after an initial busy condition, given that the requesting device is not contributing to the unit's utilization.

The miss probability obtained assuming that requests arrive at random with respect to the status of the unit is simply the stationary limit, independent of initial conditions, of this time-dependent probability. The importance of this transient effect must depend on the ratio of transfer to rotation times. If the rotation time is much greater than the transfer times, the transients should have dissipated and the reconnection request should "see" the stationary miss probability. If, on the other hand, the rotation period is short as compared to transfer times, transients should dominate. In the limit, with infinitely fast rotation speed, the probability of finding the unit busy again would be one.

Assume for the moment that transfer times follow the exponential distribution with mean t_t. Denote by q the probability that a transfer having caused a missed reconnection is still in progress after a full revolution. We have

$$q = \exp(-t_r/t_t). \tag{2.14}$$

We propose to use q as an estimate for how fast the transients dissipate. Assuming that first reconnection requests do arrive at random, this yields for the probability of experiencing a missed reconnection on a second attempt

$$p_{m_2} \simeq q + (1 - q)p_{m_1}. \tag{2.15}$$

With exponentially distributed transfers and Poisson request arrivals, subsequent attempts would experience the same miss probability as the second attempt. It is interesting to note that in a loss system with exponential service times and Poisson arrivals the probability that a request finds the unit busy again t_r time units after a previous busy condition is given by

$$p^*_{m_2} = q^* + (1 - q^*)p_{m_1} \tag{2.16}$$

where

$$q^* = \exp[-t_r/(t_t(1 - p_{m_1}))].$$

We believe this is a confirmation of our view of the phenomenon. In practice, (2.15) yields better results than (2.16), probably because the distributional assumptions under which (2.16) has been obtained are not met.

It has been noted in the literature [17], and confirmed by the author's observations (see the Appendix), that the average number of missed reconnections per transfer is fairly robust with respect to distributions of seeks and transfers. Hence, we use (2.15) also for nonexponentially distributed transfer times. Note that we are not implying that the higher miss probability on subsequent reconnection attempts is necessarily due to the same transfer that caused a previous miss. We simply use q as an estimate for the probability that the initially known busy condition is in effect after a full disk rotation. This approach, while still empirical, has the advantage of incorporating the dependence on the device revolution period.

From (2.15) one readily obtains for the overall miss probability, denoted by p_m,

$$p_m = p_{m_1}/(1 - p_{m_2} + p_{m_1}),$$

which may be rewritten as

$$p_m = p_{m_1}/[1 - q(1 - p_{m_1})]. \tag{2.17}$$

From (2.17) we see that p_m is in general greater than p_{m_1}. The average number of missed reconnections per successful I/O requests is given by

$$n_m = p_m/(1 - p_m),$$

which may be expressed as

$$n_m = n_{m_1}/(1 - q) \tag{2.18}$$

where $n_{m_1} = p_{m_1}/(1 - p_{m_1})$ is the average number of missed reconnections computed without the correction.

The average number of missed reconnections per successful I/O is thus expressed in terms of n_{m_1} and a simple corrective factor $1/(1 - q)$. It is apparent from (2.14) that the impact of this factor decreases rapidly with the ratio t_r/t_t.

2) First Reconnection Attempt: The second phenomenon, closely related to the one just discussed, concerns the miss probability experienced on a first reconnection attempt. As already mentioned, it is usually assumed [just as we did in (2.15)] that such an attempt arrives at random with respect to system state. This amounts to neglecting the fact that in most systems the I/O transfer path has to be free in order to forward the sector number to the device. In fact, the first attempt "sees" a transient unit busy probability after a latency time with an initial "path free" condition. The consensus to neglect the effects of command issuance seems to have originated from [16] where it is argued that commands have a negligible impact on total I/O time. We note that neglecting the fact that the path is free when the latency starts may result in considerable overestimation of the missed reconnection delay (the effect is more pronounced for fast spinning devices). An example is given in the Appendix. With this caveat, in the simulation results presented in the sequel, latency is allowed to start independently of the status of the I/O path. Hence, when comparing analytical results with simulation, we are assessing the accuracy of an approximate solution of a model of disk operation at a given level of abstraction. The adequacy of the level of abstraction itself can only be assessed through comparison with measurement results. (Note that the recently introduced IBM 3880 control unit can start the latency without a channel connection.)

3) Overheads and Reconnection Window: As a last point in this section, we note that in reality reconnection is requested some time (a few sectors) before the required sector is reached. This gives the time to the units within the transfer path to exchange appropriate signals. We refer to the interval between the moment reconnection is requested and the moment the correct sector is reached as the reconnection window. The latter seems to be on the order of 0.5 ms. For older disks and most applications this is negligible in comparison to the average transfer time. However, with higher transfer rates, and also for some important applications such as airlines seat reservation systems, the transfer times may be on the same order of magnitude as the reconnection window and other protocol overheads.

Note that through the use of different completion rates for various blocking points our model of Section II-B allows the inclusion of overheads for I/O path elements. While most overheads affect the missed reconnection delay by increasing

the I/O path utilization "seen" by reconnection requests, the effect of the reconnection window appears more intricate.

It is clear that some load-dependent fraction of the reconnection window is added to the transfer time on every transfer, thus resulting in higher I/O path utilization than if the reconnection window was zero. In most systems, this higher utilization is apparent to a CPU attempting to initiate an I/O operation, and causes a correspondingly increased number of rejected I/O initiation attempts. The effect of the window on missed reconnections is more difficult to assess. As the window width increases, a disk requesting reconnection finds the unit busy more often at the beginning of the window. It has, however, a greater chance that the busy unit will become available within the window duration.

An analytical and simulation study of a single blocking point (see the Appendix for details) indicates that the reconnection window may have a considerable effect on I/O path utilizations, but little, if any, on missed reconnection probability. A practical conclusion from these results is that the reconnection window need not be included when evaluating missed reconnection delays. It has, however, to be taken into account, for small transfer times, to correctly assess I/O path utilizations and related quantities such as redriven I/O's. Our analytical model provides a good estimate of the average unit busy period, and hence utilization, for a system with a single blocking point.

D. Numerical Results

We have run a number of discrete-event simulations to check the accuracy of the analysis presented. As a whole, the agreement between analytical and simulation results for the number of missed reconnections per successful I/O is good. The accuracy of the analytical results tends to be excellent for medium and large transfer times (say, 8 ms and over), and slightly less so for shorter average transfer times. There is no systematic degradation of the accuracy of the results as the I/O rates are increased.

Fig. 2 shows an example of the results obtained for a configuration with three channels and three single-path strings of eight DASD's (actuators) each, connected a control unit with two routes. The device parameters indicated on the figure correspond to the IBM 3350 device with transfers of average length of 8 kbytes. The seek time has been arbitrarily set to 70 percent of the manufacturer specified average value, and thus corresponds to a situation where 30 percent of seeks find the arm already on track (zero seeks). Discrete-event simulation results have been given for comparison. The distributional assumptions used in this simulation run are: total of seek and latency—uniform, transfer—exponential. Each simulation point corresponds to 100 000 transfer completions. The agreement between analytical and simulation results is more than fair.

E. Total DASD Service Time

For completeness, we now consider the derivation of the total I/O service time. The approach used is essentially that of [2].

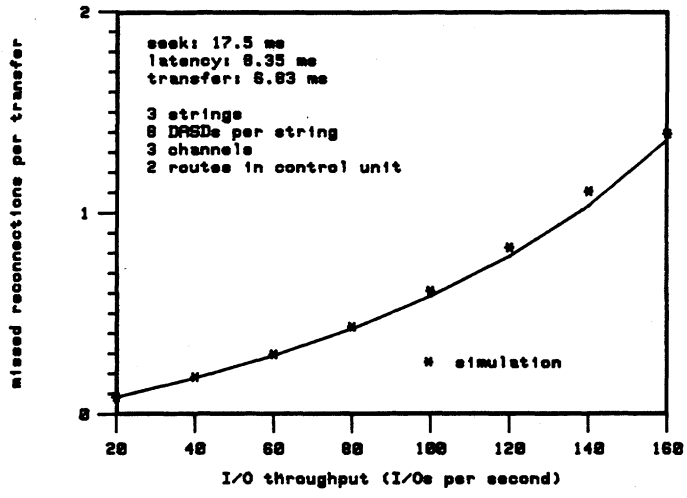


Fig. 2. Example of numerical results.

The total I/O service time for a DASD may be approximately derived as follows. Assuming I/O requests for a disk form a Poisson process and the values of basic service times are independent and identically distributed variables, we view each device as an $M/G/1$ queue. Its arrival rate is denoted by λ_d, with $\lambda_d = \lambda/(N_s N_d)$ in our case, and the average service time is the basic I/O time W_b (2.13). Denote by v_s, v_l, v_m and v_t the variance of the seek time, of the latency time, of the missed reconnection delay and that of the transfer time. Making the approximation that the above components of the disk service time are independent, we obtain for the total variance in our $M/G/1$ queue, v_b,

$$v_b \simeq v_s + v_l + v_m + v_t.$$

We assume that the variance of the seek time and of the transfer time are given. Latency times are customarily taken to be uniformly distributed between 0 and t_r, the revolution time for the disk. Hence,

$$v_l = t_r^2/12. \tag{2.19}$$

The remaining variance of the missed reconnection delay is readily obtained as

$$v_m = t_r^2 p_{m_1}(1 + p_{m_2} - p_{m_1})/(1 - p_{m_2})^2 \tag{2.20}$$

where p_{m_1} and p_{m_2} are miss probabilities on first and consecutive reconnection attempts, respectively. For the system considered they are given by (2.11) and (2.15).

Denote by c_b the coefficient of variation of the DASD service time, and by $\bar{n}_t$ the average number of I/O requests for the DASD (including the one in service).

We have

$$c_b = \sqrt{v_b}/W_b$$

and hence, by virtue of the Pollaczek–Khintchine formula [12]

$$\bar{n}_t = \rho_b + \rho_b^2(1 + c_b^2)/[2(1 - \rho_b)] \tag{2.21}$$

where

$$\rho_b = \lambda_d W_b.$$

The total I/O service time, denoted by W_t, is then obtained from Little's formula [13]

$$W_t = \bar{n}_t/\lambda_d. \tag{2.22}$$

Before closing this subject, let us note that [9] proposes a more accurate approach, taking into account the correlation between seeks.

In [4], we relax the assumption that the I/O traffic is evenly spread across strings of disks connected to a control unit and DASD's in a string. We also allow each DASD to possess its own set of characteristics of seek, latency and transfer times. Our approach is an extension of that used in Section II-B, and it proceeds as follows. First, we determine the stationary distributions of the numbers of busy paths at strings, and of the number of busy routes within the control unit. These blocking points are analyzed using loss-system models, and a simple model to estimate conditional probabilities of particular configurations of devices, strings, and channels busy given total numbers of units active. Then, missed reconnection probability is computed by analyzing numbers of requests which arrive to find various string, control unit, and channel states. The interested reader is referred to [4] for details and an example of application.

In a general case, the above solution procedure may involve a number of iterative solutions of small systems of equations. Although we do not have a theoretical convergence proof, it has been our experience that the iterative solutions involved tend to converge within just a few iterations.

In the next section, we consider the application of our approach to the analysis of present-day configurations with shared strings, shared channels and control units, and also disks with dual ports. This gives us the opportunity to compare our method with other published approaches to DASD analysis.

III. Shared Strings, Shared Channels, Dual Ports

This section is devoted to the application of our approach to systems in which strings of DASD's are shared by several control units (referred to as string switching), channels are connected to several control units, and also, systems in which disks have two ports (referred to as cross-call). Since we do not believe that it makes much sense to have shared channels or string switching with multiple paths or routes, we restrict the scope of this section to "classical" string controllers and control units with a single data transfer path. This will give us the opportunity to discuss the relationship between our method and other approaches to the analysis of disks on block-multiplexor channels.

As a first example, consider a system of two control units sharing N_s identical strings with N_d actuators per string. Each control unit has a single channel connected to it (Fig. 3). For simplicity, we assume balanced I/O traffic distribution across control units and devices. This system with string switching can be easily analyzed using our loss-system model approach.

Denote by U_s the utilization of a single string. The probability that a DASD request finds its string controller free, denoted by f_s, is

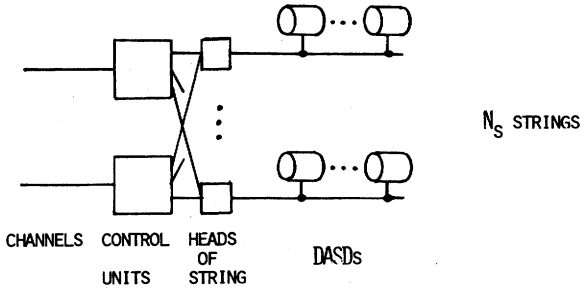


Fig. 3. System with string switching.

$$f_s = \frac{(1-U_s)N_d}{(1-U_s)N_d + U_s(N-1)} = \frac{1-U_s}{1-U_s/N_d}. \tag{3.1}$$

Let $p_u(k)$ be the stationary probability that k control units are active, $k = 0, 1, 2$. Denote by λ the rate at which an idle string requests reconnection to a control unit. An I/O is bound to the control unit on which it was initiated, so that we readily obtain as the rate of control unit activations when k control units are busy

$$\lambda(N_s - k)a_k, \text{ where } a_0 = 1 \text{ and } a_1 = \tfrac{1}{2}. \tag{3.2}$$

Hence,

$$p_u(k) = G\,\rho^k \prod_{i=1}^{k} (N_s - i + 1)a_{i-1}/i \tag{3.3}$$

where G is a normalization constant. As in Section II, ρ can be determined by bisection from the condition

$$\sum_{k=1}^{2} p_u(k)k = 2U_u \tag{3.4}$$

where U_u is the utilization of the control unit.

The probability of finding the appropriate control unit free, denoted by f_u, is given by

$$f_u = P_u(0) + \tfrac{1}{2}P_u(1) \tag{3.5}$$

where

$$P_u(k) = (N_s - k)p_u(k) \Big/ \sum_{i=0}^{2} (N_s - i)p_u(i).$$

Hence, the probability of a missed reconnection on a first attempt is

$$p_{m_1} = 1 - f_s f_u$$

and for subsequent attempts we use the correction (2.15).

The above approach may be termed "global" in that we have determined explicitly the joint probability of the state of the two control units in our system. For many systems, a "local" approach, derived from the same basic idea of using a loss-system model, is also possible. In the system of Fig. 3, consider a string and a given control unit. The probability that the string will find the control unit free upon reconnection request, f_u^*, may be expressed as

$$f_u^* = \frac{\text{rate of string requests arriving when unit free}}{\text{total rate of requests}}. \tag{3.6}$$

Let us examine the rates of request arrivals when the given control unit is busy and free, which we denote by r_b and r_f, respectively. We have up to a proportionality constant,

$$r_b = U_u \text{ Prob}\{\text{string free}|\text{control unit busy}\}$$

and

$$r_f = (1 - U_u) \text{ Prob}\{\text{string free}|\text{control unit free}\}. \tag{3.7}$$

The local approach will work if we have a means of estimating the probability that the requestor is free given the state of the requested unit. Let

x_f = Prob {string not busy with the other unit|control unit free}

and

x_b = Prob {string not busy with other unit|control unit busy with another string}.

Denote by s the fraction of control unit utilization contributed by the string (in our case, $s = 1/N_s$). We have

$$r_b = U_u(1 - s)x_b, \text{ and } r_f = (1 - U_u)x_f. \tag{3.8}$$

f_u^* is given by $r_f/(r_b + r_f)$. We do not know the values of the additional probabilities, but we note that only their relative values matter in order to obtain f_u^*. We shall assume that the state of the requested control unit has little influence on the occupation of the string with the other unit. This independence assumption yields

$$x_b \simeq x_f, \tag{3.9}$$

and, hence,

$$f_u^* \simeq (1 - U_u)/(1 - sU_u). \tag{3.10}$$

Formula (3.10) may be interpreted as meaning that the string "sees" the total utilization contributed by other strings as if this contribution took place during the time the string itself does not contribute to the control unit's utilization. It is quite remarkable that the global and local approaches yield extremely close results.

The local approach applies equally easily to other systems. As an example, consider the DASD subsystem configuration depicted in Fig. 4. The system comprises two loosely coupled CPU's sharing the whole set of devices. Every channel is connected to two control units, and every control unit has two channels—one from each CPU—hooked to it. A single string of L DASD's is connected to every control unit. The CPU's spread their load evenly across all devices, but do not necessarily generate the same I/O traffic. Thus, let p_1 and p_2 be the fractions of total I/O traffic attributable to either CPU. In order to obtain the missed reconnection delay for a DASD, it suffices to analyze the simple generic subsystem represented in Fig. 5. An independence assumption analogous to (3.9) yields the simple solution given in [5].

Fig. 6 illustrates the numerical results obtained for the above model. We have plotted the expected number of missed reconnections per transfer versus the I/O rate of a single control unit. The model parameters correspond to a configuration of 8 actuators per string. The device rotation time is 17 ms, and the average transfer time is 10 ms. The traffic split between CPU's (and channels) is 2 to 1, i.e., we have $p_1 = 2/3$ and $p_2 = 1/2$. We have also represented the results of a discrete-event simulation of the system as well as the values which would be obtained without our correction (2.15) for nongeometrically distributed numbers of missed reconnections per transfer. The simulation assumes that both the total of seek and latency, and transfer times, are exponentially distributed. Each simulation

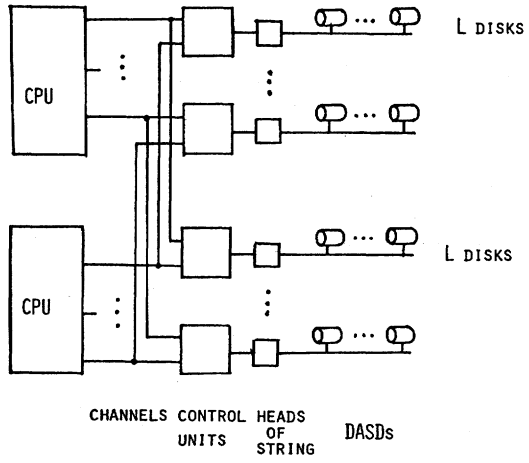


Fig. 4. Two CPU's sharing a DASD subsystem.

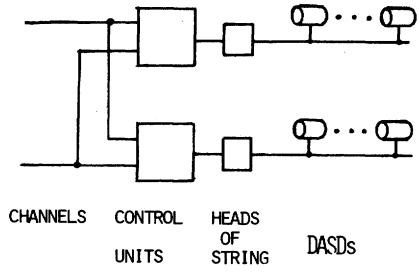


Fig. 5. Generic subsystem for analysis.

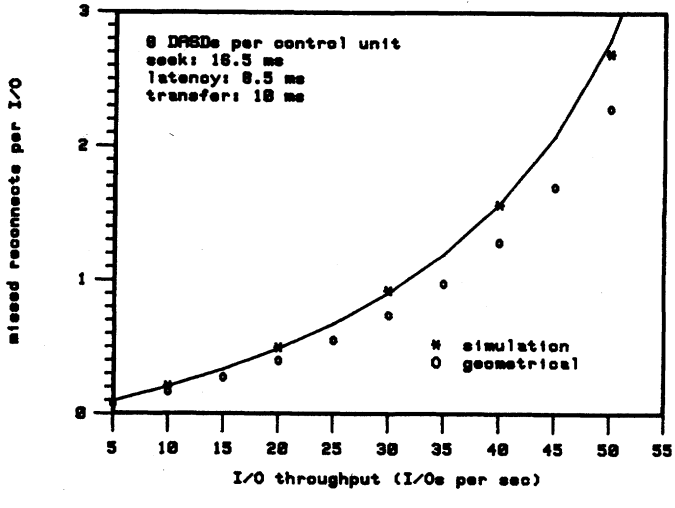


Fig. 6. Comparison of analytical and simulation results.

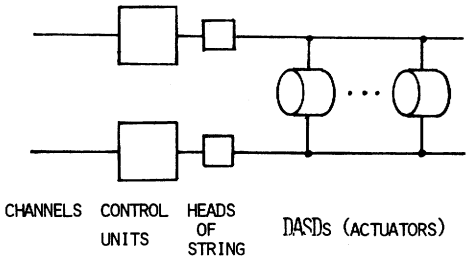


Fig. 7. System with cross-call.

point corresponds to 100 000 transfer completions. Note that the agreement between analytical results and simulation in Figure 6 holds well into prohibitively high loads and average numbers of missed reconnections per successful I/O.

Our last example in this section is a set of N_d disks (actuators) with dual ports, connected to two string controllers. Each string controller is connected to a single control unit and a single channel, as shown in Fig. 7. We consider here this system as a dual-pathing arrangement for access from the same CPU. For simplicity, we assume a homogeneous and balanced traffic across all units. We also assume that an I/O operation which starts on a given access path (channel, control unit, string controller) is bound to that path until completion. Such "cross-call" arrangements are offered by several DASD manufacturers.

The global approach for this system is quite straightforward. Let $p(j)$ be the probability that j disks are simultaneously involved in transfer, $j = 0, 1, 2$. We have

$$p(j) = G\,\rho^j \left[\prod_{i=1}^{j} (N_d - i + 1)a_{i-1}\right] \Big/ j! \qquad (3.11)$$

where G is a normalization constant, and

$$a_0 = 1, \text{ and } a_1 = \tfrac{1}{2}. \qquad (3.12)$$

ρ is determined from the condition

$$\sum_{j=1}^{2} p(j)j = 2U_s \qquad (3.13)$$

where U_s is the utilization of a single string. The probability of a missed reconnection on a first attempt, p_{m_1}, is readily obtained as

$$p_{m_1} = P(1)a_1 + P(2) \qquad (3.14)$$

where

$$P(j) = (N_d - j)p(j) \Big/ \sum_{i=0}^{2} (N_d - i)p(i).$$

For the local approach, we need the probability that an actuator is free given the state of a particular string. Assuming that the probability of the actuator not being busy with the other string controller is independent of the state of the tagged controller, we immediately get

$$p_{m_1} = (1 - U_s)/(1 - U_s/N_d). \qquad (3.15)$$

Again, the results of local and global approach are very close.

It is interesting to note that, as long as an I/O is bound to the channel on which it has been initiated, the cross-call arrangement of Fig. 7 is functionally equivalent to a single string with two paths, connected to a single control unit with two routes and two channels.

Both approaches are based on the loss-system model. The global approach is well suited to the study of multipath systems in the sense of Section II. It can be extended, at the expense of increased complexity, to imbalanced systems with several classes of requests. The local approach, on the other hand, is quite simpler when applied to imbalanced configurations, and has also no trouble handling various systems with shared units. It does not appear, however, to be well suited to the general case of multiple, nondedicated transfer paths.

To the best of the author's knowledge, the number of publications dealing specifically with DASD's on block-multiplexor channels is quite limited (cf. [11]), and none of them considers multiple transfer paths in the sense of Section II. The equivalence and decomposition approach of [3] and the method of surrogate delays [10] are designed for use with closed queueing network models, and, therefore, do not lend themselves to a simple comparison with our method. Zahorjan *et al.* [17] and Bard [2] both deal, as does this paper, with directly estimating the missed reconnection probability. In Section II, we have discussed the differences in the corrections proposed to account for the miss probability on second and subsequent reconnection attempts. Here, we concentrate on the analysis of first attempts. The work by Zahorjan *et al.* is limited to a single blocking point. For such systems, the results of [17], [2] and both our global and loal approaches are all identical.

For systems with multiple blocking points, Bard's approach [2], albeit also throughput driven, is in spirit quite different from ours. He derives a set of equations for occupation probabilities of particular transfer paths with a particular device. As this set involves many more unknowns than there are equations, Bard chooses among all feasible solutions the one which maximizes a quantity called entropy. While this approach does allow "cranking out a solution" automatically, it does not, in the author's view, enhance one's intuition and understanding of the phenomena involved. Nor does it show what simplifying assumptions are introduced in the solution process. On the other hand, our approach uses a set of loss-system models with well spelled-out assumptions.

The probability of a missed reconnection is given in [2] by a formula involving a number of joint element utilizations (14). When applied to our example with string switching of Fig. 3, or to the system with shared channels of Fig. 5, this formula becomes identical to those yielded by our local analysis. Thus, for these systems, the maximum entropy principle appears to implicitly introduce independence assumptions of the type of (3.9). Unlike the maximum entropy principle, however, an independence assumption is essentially operational in nature (in the sense of operational analysis [7]) in that, when transposed into the operational domain, it relates readily measureable quantities, and can easily be tested.

Formulas like (3.10), simple and appealing, have actually been in use in the industry for some time, apparently without much theoretical justification. [8] reports on measurement campaign to validate their results. Finally, note that blind application of these formulas is not always possible. The system

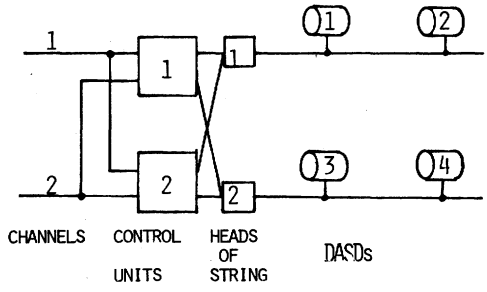


Fig. 8. System with string switching and shared channels.

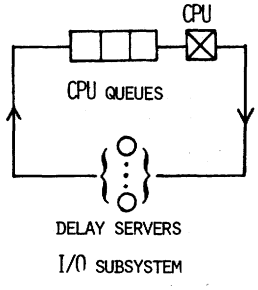


Fig. 9. CPU-I/O model.

of Fig. 8, borrowed from [2], is a good example. Here, when one considers the reconnection of a control unit to a channel, the knowledge of the idle state of the string considerably affects the probability of the control unit being free. We show in the Appendix how the local approach can be applied to this system. The results obtained differ slightly from those of [2].

In the next section, we briefly discuss the inclusion of our DASD subsystem model in a larger system model, and also attempt a comparison with the equivalence and decomposition approach of [3].

Inclusion in a System Model

The analysis of DASD subsystems considered in preceding sections yields the basic I/O service time for a disk at a given rate of successful I/O's directed to that device. A simple way of embedding such an analysis in a larger system model is by replacing the DASD subsystem by a set of delay servers, each delay being equal to the average total DASD time (including queueing). The latter may be approximately computed for every DASD using the Pollaczek–Khintchine formula [12] (cf. Section II), or the corresponding finite queueing room analysis when the load conditions make it necessary. The I/O throughput is then determined iteratively by analyzing a queueing network of the type shown in Fig. 9. When convergence has been reached, usually within just a few iterations, both CPU utilization and I/O service times are known. The approach described is not new and has been used by other authors, e.g., [1]. A somewhat more sophisticated approach is proposed in [4].

In a previous paper [3], the author considered a different approach to DASD subsystems, specifically designed for use with the equivalence and decomposition analysis of queueing models. It is interesting to apply the two methods to a system model, and compare their results. For the comparison we

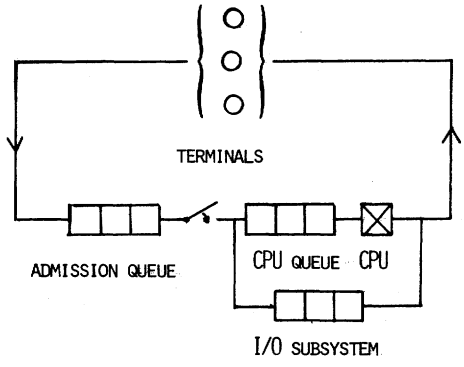


Fig. 10. Queueing model of an interactive system.

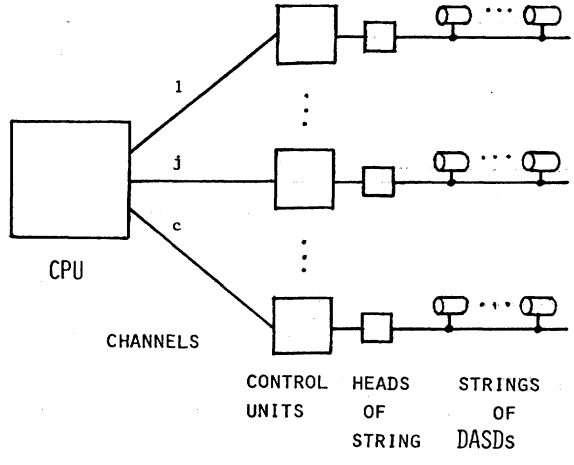


Fig. 11. DASD subsystem for comparison of approaches.

choose the simple system depicted in Fig. 10. It represents an interactive multiprogramming system. The memory queue holds commands (jobs) awaiting admission into the multiprogramming set. The number of jobs admitted is not allowed to exceed a fixed maximum degree of multiprogramming. Upon entry into the multiprogramming set commands join the CPU queue. The only I/O devices represented are disks, and the DASD subsystem is taken to have the simple structure shown in Fig. 11, viz. a single string of disks per control unit, each connected to a single channel. A DASD reconnection request has thus a single blocking point. We consider the above model under the assumption that all the I/O requests are statistically identical with respect to their seek, latency and transfer times.

The I/O subsystem described corresponds to the basic DASD model of [3] and its equivalence and decomposition analysis may be found there. Assuming exponentially distributed CPU service times, it is not difficult to obtain an approximate solution to the overall model of Fig. 10.

The throughput-driven analysis of the system proceeds as follows. Using the results of Section II we readily obtain the basic I/O service time of a DASD, W_b. Hence, the total I/O service time (including queueing), W_t is approximately computed using the Pollaczek–Khintchine formula, as described in Section II. In the iterative analysis of the multiprogramming set model we take the simplest representation whereby the

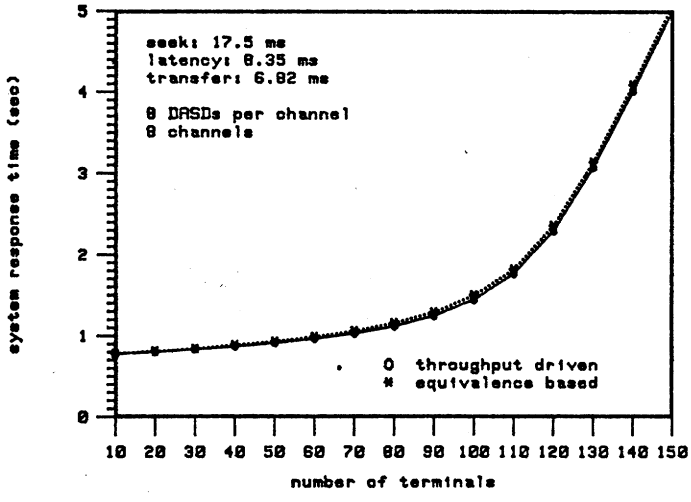


Fig. 12. Comparison of the two approaches.

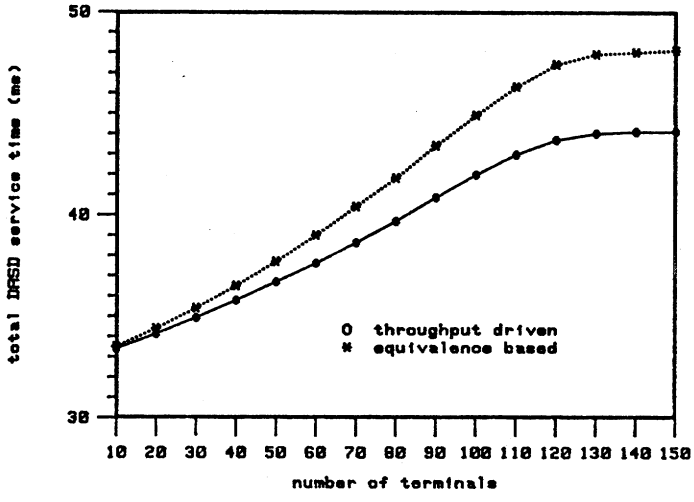


Fig. 13. Comparison of the two approaches.

DASD subsystem is viewed as a set of delay servers with service time W_t.

Numerical results obtained from both approaches are shown in Fig. 12. We have plotted there the expected system response time (time a command spends in the system)—as predicted by the two approaches, versus the total number of terminals active in the system. The following values were used for model parameters:

average user think time: 10 s
average CPU time per command: 100 ms
average number of I/O requests per command: 20
maximum multiprogramming degree: 20

The DASD subsystem comprises eight channels, two of which equally share 50 percent of the I/O traffic, the other 50% being evenly distributed across the remaining channels. All the disks at a given channel are assumed to be equally utilized.

We observe that both methods yield remarkably close results for the average system response time. It is interesting to note that these results are obtained despite differences of a few percent in total I/O service times as illustrated in Fig. 13. As regards computational time and space requirements, the throughput-driven approach appears to be superior. This statement must, however, be moderated. Due to the iterative nature of the throughput-driven approach the execution time may vary depending on the particular set of parameters. Also, the equivalence based approach lends itself nicely to parametric studies (e.g., varying the number of terminals), while this is not necessarily the case for the other method.

V. Conclusion

We have presented a method for analyzing disk subsystems with block-multiplexor channels. This method takes the throughput of successful disk I/O requests as basis. The seek, latency and transfer times are assumed known, and the goal of the analysis is to determine the delay due to missed device reconnections. The latter may occur at elements of the I/O transfer path which are shared by several DASD's. Such elements are referred to as blocking points, and include channels, routes (within control units) and paths (within strings of disks). In the approach presented, conflicts at each of these blocking points are analyzed separately using essentially a loss-system model in which units demanding reconnection are viewed as sources of requests for the resources of the blocking point.

The analysis is introduced for a model with multiple transfer paths in control units and strings of DASD's. In this way a framework is provided for the evaluation of both existing configurations and those including forthcoming products with multiple routes and paths (cf. [15]).

Iteration may be involved in the solution procedure when multiple routes or paths are present. Usually, configurations with a single route per control unit and a single path per string of disks—preponderant in current systems, do not require iteration, and lead to a particularly simple solution.

A simple correction is proposed to account for the nongeometric distribution of the number of missed reconnections experienced by an I/O request. This correction is effective for longer transfer times. For very short transfer times, the effects of the device reconnection window on both missed reconnections and I/O path busy period are discussed.

The throughput-driven nature of the method makes it directly applicable to problems where one is interested in the performance of the I/O subsystem under a specified I/O traffic. Its inclusion in a system model, however, requires an iterative approach.

Numerical results obtained with the method show a good agreement, throughout wide ranges of loads, with the results of discrete-event simulations of the examples considered.

The basic idea of the method presented, viz. that of considering blocking points separately, can be successfully applied to the study of other DASD configurations, including, for example, a cache in the control unit.

Appendix

A. Influence of Path State at Latency Start

As an example let us take the configuration considered in Fig. 2. Table I shows the average numbers of missed reconnections per transfer observed in two sets of discrete-event

TABLE I

I/O throughput (I/O's/second)	40	60	80	100	120	140	160
misses per transfer							
path free	0.124	0.212	0.323	0.461	0.635	0.885	1.198
any state	0.180	0.296	0.434	0.623	0.821	1.086	1.447

TABLE II

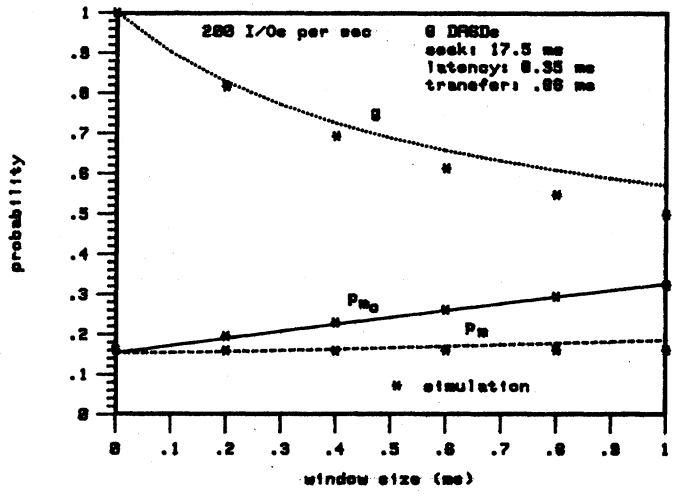
I/O throughput (I/O's/second)	40	60	80	100	120	140	160
misses per transfer							
path free	0.122	0.210	0.313	0.450	0.630	0.873	1.160
any state	0.190	0.308	0.452	0.623	0.826	1.108	1.499

simulations of this system. In the first set, latency is allowed to start only if the I/O path is free. In the second set, latency starts independently of the status of the I/O path. The distributional assumptions are: transfer times—exponential, seek times—uniform, latency—uniform, for both sets of simulations. Each simulation point corresponds to 100,000 transfer completions. Table II shows the results obtained for a device with a shorter rotation period of 15 ms (versus 16.7 ms in Table I). We observe that neglecting the status of the I/O path at latency start can result in an overestimation of the missed reconnection delay by up to 40 percent. As expected, the overestimation is more pronounced for the faster spinning device.

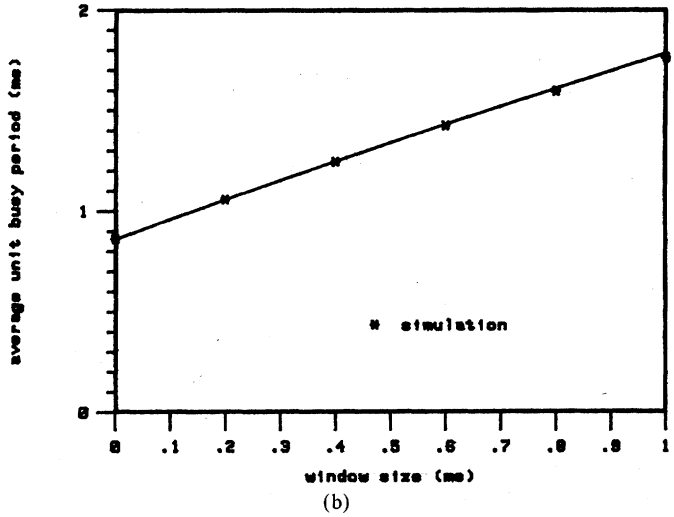
B. Reconnection Window

The interested reader will find in [4] the description and derivation details of our model of the reconnection window. Numerical results obtained from this model are exemplified in Fig. 14 for an average transfer time of 0.86 ms. The rotation period, t_r, is set to 16.7 ms. We have plotted the probabilities p_{m0}: unit busy at beginning of window, g: busy unit doesn't become available and p_m: missed reconnection, as well as the average unit busy period, versus the reconnection window size. For comparison, the results of discrete-event simulation with exponentially distributed DASD transfer times are also included. Each simulation point corresponds to 500 000 transfer completions.

It is interesting to note that, within the range of values explored, the window size has little influence on the missed reconnection probability. The agreement between analytical and simulation results is good as long as the window size is not too large compared with the transfer time. The average unit busy period exhibits a remarkable robustness, even for relatively large window sizes.



(a)



(b)

Fig. 14. (a) Effect of reconnection window. (b) Effect of reconnection window.

C. Application to the Example of Fig. 8

Consider the system represented in Fig. 8. It consists of two control units sharing two channels and two strings of two DASD's each. Let us apply the local approach to the reconnection of device 1 along the path composed of head of string 1, control unit 1, and channel 1. For the reconnection of the device to its string, and for the reconnection of the string to the control unit, we use independence assumptions analogous to (3.9). Hence, we get for the corresponding probabilities of finding the blocking point free

$$f_s = \frac{1 - \langle S_1 \rangle}{1 - \langle D_1 \rangle}$$

and

$$f_u = \frac{1 - \langle U_1 \rangle}{1 - \langle U_1 S_1 \rangle}$$

where, following the notations of [2], $\langle S_1 \rangle$ is the utilization of string 1, $\langle D_1 \rangle$ is the contribution of device 1 to that utilization, $\langle U_1 \rangle$ is the utilization of control unit 1, and $\langle U_1 S_1 \rangle$ is the contribution of string 1 to the latter utilization.

The probability of the control unit finding the channel free may be expressed as

$$f_c = 1 - \frac{A \cdot x_b}{A \cdot x_b + B \cdot x_f}$$

where

A = Prob {channel 1 busy | string 1 idle},

B = Prob {channel 1 idle | string 1 idle},

x_b = Prob {control unit 1 idle|channel 1 busy, string 1 idle},
x_f = Prob {control unit 1 idle|channel 1 idle, string 1 idle}.
Note that we cannot assume that the activity of the control unit with channel 2 is independent of the status of channel 1. Indeed, since string 1 is known to be idle, the control unit cannot be active with channel 2 when channel 1 is active. Therefore, we analyze the probabilities x_b and x_f in more detail. We have from the definition of conditional probabilities

$$x_b = \frac{A \cdot \text{Prob}\{\text{string 1 idle}\} - \text{Prob}\{\text{control unit 1 and chnl 1 busy, string 1 idle}\}}{A \cdot \text{Prob}\{\text{string 1 idle}\}}$$

and

$$x_f = \frac{B \cdot \text{Prob}\{\text{string 1 idle}\} - \text{Prob}\{\text{control unit 1 and chnl 2 busy, string 1 idle}\}}{B \cdot \text{Prob}\{\text{string 1 idle}\}}.$$

Hence, we readily get

$$f_c = 1 - \frac{A - \text{Prob}\{\text{control unit 1 and channel 1 busy}|\text{string 1 idle}\}}{1 - \text{Prob}\{\text{control unit 1 busy}|\text{string 1 idle}\}}.$$

The conditional probabilities in the above expression can all be estimated using independence assumptions which yield

$$A = \frac{\langle C_1 \rangle - \langle C_1 S_1 \rangle}{1 - \langle C_1 S_1 \rangle},$$

$$\text{Prob}\{\text{control unit 1 and channel 1 busy}|\text{string 1 idle}\} = \frac{\langle U_1 C_1 \rangle - \langle U_1 C_1 S_1 \rangle}{1 - \langle U_1 C_1 S_1 \rangle},$$

$$\text{Prob}\{\text{control unit 1 busy}|\text{string 1 idle}\} = \frac{\langle U_1 \rangle - \langle U_1 S_1 \rangle}{1 - \langle U_1 S_1 \rangle}$$

where $\langle U_1 C_1 S_1 \rangle$ denotes the utilization of channel 1 contributed by string 1 via control unit 1, $\langle C_1 \rangle$ is the utilization of channel 1, and $\langle C_1 S_1 \rangle$ is the contribution of string 1 to that utilization. Using the data of [2], we obtain

$$f_s = 0.592, \quad f_u = 0.796, \quad \text{and } f_c = 0.884,$$

i.e., a successful reconnection probability of 0.416 which is not very different from the 0.419 obtained from formula (14) in [2].

D. Influence of Transfer Time Distribution and Simulation Accuracy

As an example we show in Table III discrete-event simulation results for the configuration of Fig. 2 and constant transfer times. These results are directly comparable with those of Table I, line "any state." In order to give an idea of the accuracy of the simulation results, we also give 90 percent level approximate confidence intervals estimated from a set of independent runs.

TABLE III

I/O throughput (I/O's/second)	40	60	80	100	120	140	160
misses per transfer any state	0.168	0.283	0.412	0.584	0.801	1.092	1.456
confidence interval	0.169	0.279	0.413	0.582	0.793	1.077	1.428
	0.171	0.285	0.422	0.592	0.805	1.103	1.451

REFERENCES

[1] Y. Bard, "The VM/370 performance predictor," *Comput. Surveys*, vol. 10, pp. 333–342, 1978.

[2] ——, "A model of shared DASD and multipathing," *Commun. Ass. Comput. Mach.*, vol. 23 pp. 564–572, 1980.

[3] A. Brandwajn, "Models of DASD subsystems: Basic model of reconnection," *Perform. Eval.*, vol. 1, pp. 263–281, 1981.

[4] ——, "Models of DASD subsystems: multiple access paths—A throughput-driven approach," Amdahl Corp., Sunnyvale, CA, Amdahl Tech. Rep., P/N 820457-700B, available upon request.

[5] ——, "A capacity planning model of a DASD subsystem," in *Performance '81*, F. J. Kylstra Ed. Amsterdam, The Netherlands: North-Holland, Nov. 1981, pp. 401–413.

[6] R. B. Cooper, *Introduction to Queueing Theory*. New York: Macmillan, 1972.

[7] P. J. Denning and J. P. Buzen, "The operational analysis of queueing network models," *Comput. Surveys*, vol. 10, pp. 225–241.

[8] H. Hoffman, "DASD configuration analysis," in *Proc. CMG X Int. Conf.*, Dallas, TX, 1979, pp. 285–302.

[9] M. Hofri, "Disk scheduling: FCFS versus SSTF revisited," *Commun. Ass. Comput. Mach.*, vol. 23, pp. 645–653, 1980.

[10] P. Jacobson and E. W. Lazowska, "The method of surrogate delays: Simultaneous resource possession in analytic models of computer systems," *Perform. Eval. Rev.*, vol. 10, pp. 165–174, 1981.

[11] M. G. Kienzle and K. C. Sevcik, "Survey of analytic queueing network models of computer systems," in *Proc. Conf. Simulation, Measurement, Model. Comput. Syst.*, Boulder, CO, 1979; also in *Perform. Eval. Rev.*, vol. 8, pp. 113–129, 1979.

[12] L. Kleinrock, *Queueing Systems, Vol. 1: Theory*. New York: Wiley, 1975.

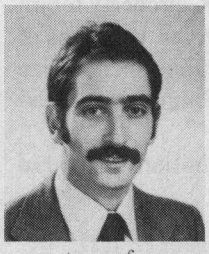
[13] J. D. Little, "A proof of the queueing formula $L = \lambda W$," *Oper. Res.*, vol. 9, pp. 383–387, 1961.

[14] A. Rafii, "Effects of channel blocking on the performance of shared disk pack in a multi-computer system," in *Proc. Conf. Simulation, Measurement, Model. Comput. Syst.*, Boulder, CO, 1979; also in *Perform. Eval. Rev.* vol. 8, pp. 83–87, 1979.

[15] T. Scannel, "CDC offers rival to IBM 3380 disk system," *Comput. World*, p. 14, Jan. 26, 1981.

[16] N. Wilhelm, "A general model for the performance of disk systems," *J. Ass. Comput. Mach.*, vol. 24, pp. 14–31, 1977.

[17] J. Zahorjan, J. N. P. Hume, and K. C. Sevcik, "A queueing model of a rotational position sensing disk system," *INFOR 16*, pp. 199–216, 1978.



Alexandre Brandwajn received the Ingénieur Civil degree from the Ecole Nationale Supérieure des Télécommunications, Paris, France, in 1971, and the Docteur-Ingénieur and Docteur ès-Sciences degrees from the University of Paris VI in 1972 and 1975, respectively.

While a Researcher at IRIA-Laboria, Rocquencourt, France, from 1971 to 1975, he worked on performance evaluation of an operating system and on solution methods for queuing models of computer performance. In 1975 he joined the Ecole Nationale Supérieure des Télécommunications where he was a Professor of Computer Science until 1979. He is currently with Amdahl Corporation, Sunnyvale, CA. His research interests include the areas of analytical and numerical methods for queueing models, dynamically adaptive computer architectures, and fault tolerance techniques.

Dr. Brandwajn is the author of a number of articles and conference presentations, and a member of the Association for Computing Machinery.