Lawrence Berkeley National Laboratory
Recent Work

Title
Performance Evaluation of Mass Storage Systems for Scientific Databases

Permalink
https://escholarship.org/uc/item/1tj6n8rx

Authors
Segev, A.
Seshadri, S.
Rotem, D.

Publication Date
1994-09-07

eScholarship.org Powered by the California Digital Library
University of California

LBL-36092
UC-405

Lawrence Berkeley Laboratory

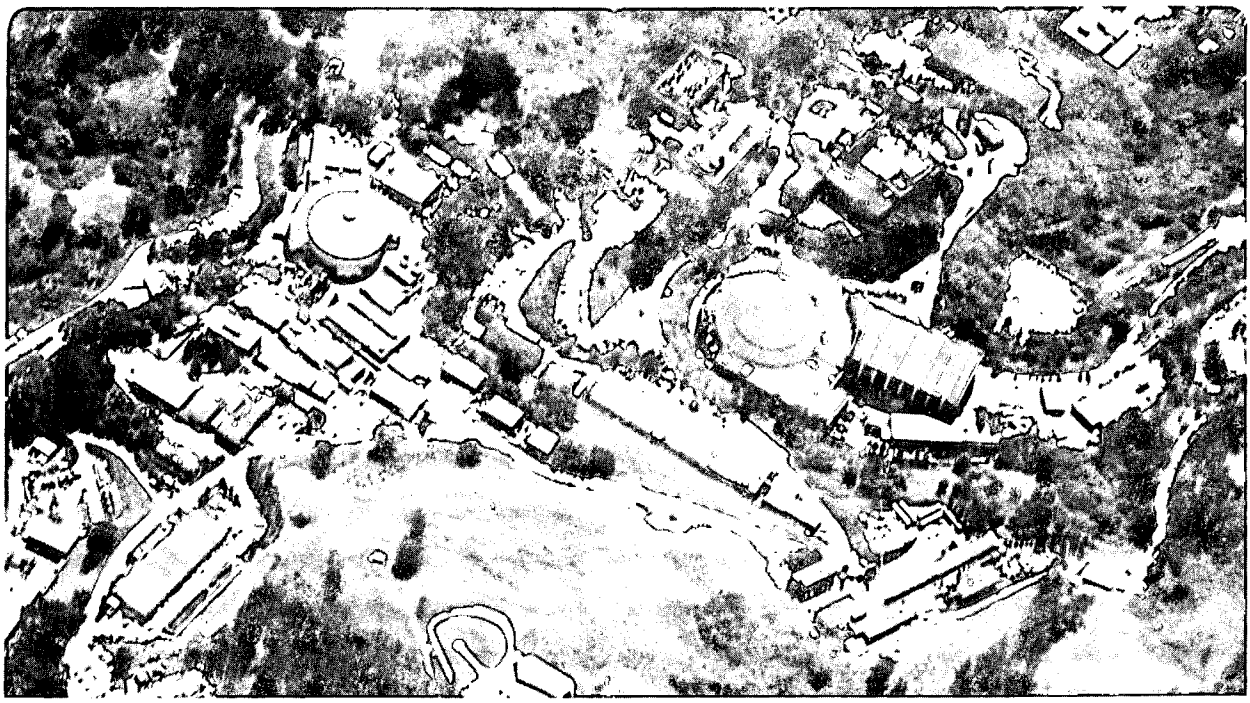
UNIVERSITY OF CALIFORNIA

Information and Computing Sciences Division

Performance Evaluation of Mass Storage Systems for Scientific Databases

A. Segev, S. Seshadri, and D. Rotem

September 1994



REFERENCE COPY
Does Not
Circulate
Bldg. 50 Library.
LBL-36092
Copy 1

Prepared for the U.S. Department of Energy under Contract Number DE-AC03-76SF00098

DISCLAIMER

This document was prepared as an account of work sponsored by the United States Government. While this document is believed to contain correct information, neither the United States Government nor any agency thereof, nor the Regents of the University of California, nor any of their employees, makes any warranty, express or implied, or assumes any legal responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by its trade name, trademark, manufacturer, or otherwise, does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency thereof, or the Regents of the University of California. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof or the Regents of the University of California.

LBL-36092

Performance Evaluation of Mass Storage Systems for Scientific Databases

Arie Segev

Haas School of Business, University of California, and
Information & Computing Sciences Division, Lawrence Berkeley Laboratory
Berkeley, CA 94720

Sridhar Seshadri,
Haas School of Business, University of California
Berkeley, CA 94720

and

Doron Rotem
Information & Computing Sciences Division, Lawrence Berkeley Laboratory
Berkeley, CA 94720

This work was supported by the Applied Mathematical Sciences Research Program of the Office of Energy, U.S. Department of Energy under Contract DE-AC03-76SF00098.

Introduction: Mass storage systems for computers are the solution to economic storage of vast volumes of data. These systems evolved from the traditional tape libraries manned by operating personnel and the automation of the storage and retrieval function has led to significant improvement in performance. But in contrast to traditional computer systems, little work has been done to characterize performance in terms of the design parameters. The design and performance analysis of mass storage systems is complicated due to several reasons. A major reason for the complexity is the time lags that may occur in retrieving parts of the information meant for the same query. The usual queuing models used for analyzing disk performance (for example see Lavenberg (1983)) are not directly applicable, because there is greater scope for working in parallel in mass storage systems such as robotic libraries, which help mitigate these shortcomings. In this note, robotic libraries are modelled as queueing systems and explicit results related to performance are obtained. The physical model corresponds to a mass storage system, where the information is stored in cassettes, which are retrieved by robots to be read using one or two read heads. The results pertain to the effect of file splitting on cassettes , and optimal configuration and control of robots that perform the retrieval and storage functions.

Robotic libraries are usually termed Automatic Tape Libraries, ATL The automation of tape libraries was in response to the growing volume of tapes that need to be stored and retrieved in a number of diverse applications. The work reported in this paper is a step further in the direction of automation. In simulating weather patterns, the volume of data generated is so large that several volumes of tape information get generated. Later when reconstructing specific portions of the simulation, these tapes have to be retrieved in an *ad hoc* fashion. So the problem of managing the *dynamic* storage and retrieval of tapes becomes important, with the ultimate goal

of reducing the mean delay per query to the ATL The ATL is referred in this note as Storage System. It may comprise one or more Read heads, where each head represents a channel of communication to the external system. There may be one or more robots that perform the retrieval and storage (together termed as fetch) function. This model conforms to some of the system configurations produced by Exabyte and of the IBM 3480 magnetic tape libraries described by Ranade. In addition the analysis applies with some restrictions to the carousel tape libraries. The tapes are termed cassettes in the paper. The usual emphasis due to the high cost of these systems is on reliability, so duplication is the norm (see Ranade). The duplication leads to some interesting issues vis-a-vis performance as retrievals can be performed in parallel and may not be in the sequence requested.

In section 1, a single robot model is examined. It is shown that neglecting the reading time in the analysis leads to the standard GI/G/1 queue model, and the error due to this approximation is shown to be less than 10% under moderate load and with considerable overlap between the reading and fetch time distributions. Formulae are derived to show the effect of file splitting on several cassettes. These formulae are intended for later use in a file allocation algorithm with the objective of minimizing the mean delay per query. As a rule of thumb the mean delay increases as a quadratic function of the split.

In section 2, we investigate the utility of having more than one robot. Bounds are derived that compare a two robot system with one that has a single but twice as fast robot. The purpose of this analysis is twofold: to compare the two systems and obtain formulae for the two robot case as it is not directly amenable to analysis. It is shown that unless the cassettes are allowed to be

read in the sequence retrieved rather than in the sequence requested, there may be blocking in the two robot system leading to poor performance. With the flexibility of reading in the sequence retrieved by the robots, the two robot system is shown to be preferable when the average number of cassettes to be retrieved per query is large. In contrast, when the load is light and the reliability of the robot high, the single fast robot is preferred.

In section 3, the model with two robots and two read heads is analyzed. It is shown under the assumption of deterministic fetch times but random number of cassettes required per query, that the function of the second read head is mainly of pre-emption value. The optimal service policy under realistic assumptions, turns out to be to serve the query with the smallest remaining work after completing all current fetch orders (which is a system requirement).

Several problems are open for investigation. First is the question of improving the approximation of ignoring the read times or bounding the performance using results related to tandem queues which is the appropriate model when reading times are significant. The second interesting problem is to extend the optimality of the retrieval policy spelt out in the last paragraph to the case of random fetch times, preferably uniformly distributed to conform to the real life system. A third interesting problem is to improve the bound on the delay performance of two robots when they work together on a query (in keeping with the policy of sequencing per the shortest remaining work rule). In a follow-up work we intend to test and use the formulae that express mean delay as a function of file splitting for determining file allocation.

1. Single Server Case: The basic assumptions are that there is a single reading head, and a single robot which retrieves one cassette at a time. Queries arrive at a rate of λ per hour. The random variable representing the inter-arrival time between the nth and the $(n+1)^{st}$ query is denoted by T_n. The robot's moves are termed Up and Down, and represented as random variables U and D. The time to travel for the ith time to the reading head is denoted as U_i and the time to return to the storage area for the ith time by D_i. The reading time from the instant the tape is brought to the head and released by the robot is R_n for the nth query. Define the delay experienced by a query as the time that elapses from the instant of initiation of the query at the read head till the reading begins. This will be the definition of initiation of a query throughout this note. The emphasis is on the delay experienced due to the service time of and congestion at the robot. Let W_n represent the delay experienced by the nth query. The recursion for the delay can be written as:

$$W_1 = U_1$$

$$W_n = \max \{ [W_{n-1} + D_{n-1} - T_{n-1}]^+ + U_n , [W_{n-1} + R_{n-1} - T_{n-1}]^+ \} \tag{1}$$

$$\geq \max \{ [W_{n-1} + D_{n-1} + U_n - T_{n-1}]^+ , [W_{n-1} + R_{n-1} - T_{n-1}]^+ \}$$

To interpret (1), a query has to wait if the previous query has not finished reading from the tape, or the robot has not returned to the storage area. This accounts for the *max* operation. If the query arrives before either of these actions are completed, then there is a delay, else there is no delay due to these components, which explains taking the positive part, i.e. $[\ .]^+$. Finally a delay due to the *Up* time has to be experienced.

In most practical situations the minimum of the Up and Down times will dominate the Read time. In particular assuming uniform distributions for travel Up and Down, and for reading as well, this statement can be verified in a given set-up. We do obtain such uniform distributions, with read times averaging 5 seconds . and Up plus Down times exceeding 15 seconds (for example see Ranade (1992)). Now continuing the recursions with this assumption:

$$W_2 = [U_1 + D_1 - T_1]^+ + U_2 = \max[0, U_1 + D_1 - T_1] + U_2$$

$$W_3 = [W_2 + D_2 - T_2]^+ + U_3 = \max[0, \max[U_2 + D_2 - T_2, U_2 + D_2 + U_1 + D_1 - T_1 - T_2]] + U_3$$

$$= \max[0, U_2 + D_2 - T_2, U_2 + D_2 + U_1 + D_1 - T_1 - T_2] + U_3$$

$$W_n = \max[0, U_{n-1} + D_{n-1} - T_{n-1}, U_{n-1} + D_{n-1} + U_{n-2} + D_{n-2} - T_{n-2} - T_{n-1}, \ldots] + U_n \qquad (2)$$

This is the Lindley recursion for the delay in the G/G/1 queue (see Wolff 1990), with a little modification in that the last term in (2) does not include the Down time.

Now we use this simple model to evaluate the effect of *file splitting*. Consider first the case when the queries arrive as per a poisson process, and each query requires just one cassette to be retrieved. Assume (for simplicity) that successive queries need different cassettes with probability one. In this case the mean delay can be written as:

$$EW = \frac{\lambda\left(EU_1^2 + ED_1^2 + 2EU_1 ED_1\right)}{2(1-\lambda(EU_1 + ED_1))} + EU_1 \qquad (3)$$

In the second case we assume that each query will need one or two cassettes. When more than one cassette is required the delay is defined as the time that elapses from the initiation of the query till the *last* tape get loaded for reading. The scheme is:

$$P\left(\textit{one}\text{ cassette required}\right) = p_1 \quad \text{and} \quad P\left(\textit{two}\text{ cassettes required}\right) = p_2$$

We can then write the delay as:

$$EW(p_1) = \frac{\lambda\left((2-p_1)(EU_1^2+ED_1^2)+2p_1EU_1ED_1+2(1-p_1)(EU_1+ED_1)^2\right)}{2(1-\lambda(EU_1+ED_1)} + EU_1 + (1-p_1)(EU_1+ED_1) \tag{4}$$

This leads to substantial increase in the expected value of the delay as seen from Fig-1. This figure has been generated using uniform distributions on [8,16] and [6,12] for the Up and Down times, and values of p_1 ranging from 0.5 to 1. For example the delay at $p_1 = 1$ more than doubles when $p_1 = 0.5$ for an arrival rate of 80 queries per hour. In particular note that the above analysis is under light to medium load conditions (ρ =0.05 - 0.70) and the increase in delay is a quadratic function of the split compared to usual nearly linear increase experienced for these load conditions in the M/G/1 model. The analysis can be easily extended to cases where the number of cassettes required is a random variable. For example, if the probability that j cassettes are required is p_j then the general expression for the mean delay as defined earlier is:

$$\begin{aligned} EW(p_1,p_2,\ldots,p_j,..) &= \lambda E(S^2)/2(1-\rho) + EU_1 + \sum_{j>1} p_j\left(jE(U_1+D_1)\right) \\ E(S^2) &= \sum_j p_j\left(jE(U_1+D_2)^2 + j(j-1)(E(U_1+D_1))^2\right) \end{aligned} \tag{5}$$

However note that the Up and Down times need not be independent of one another. It is possible that the Down time is a fraction of the Up time when there are several tapes to be retrieved. For example if the Down time is α times the succeeding Up time, then the situation certainly worsens, as the expression for the second moment reveals. In the case of independent Up and Down times the expression for :

$$E(U_1 + D_2)^2 = EU_1^2 + 2EU_1ED_1 + ED_1^2$$

But, in the latter case :

$$E(U_1 + D_2)^2 = (1+\alpha)^2 EU_1^2$$

This increase is not of much consequence in the context of uniform distributions as demonstrated later.

Several performance issues can now be investigated. Firstly, how robust is the assumption of read times being negligible. From (1) it is clear that the formulae for the mean delay given above are lower bounds. Fig-2 shows the error in the approximation obtained from simulation. In these simulations, the splitting probabilities and the arrival rate of queries are varied to obtain different operating conditions. The Up and Down times are assumed to be uniformly distributed in the interval of [8,16] and [6,12] seconds. The reading time is assumed to be uniformly distributed in [12,26]. Therefore there is a substantial overlap between the read and fetch times. (Note that throughout when comparing the performance of two or more systems, the same sequence of random numbers were used to simulate them. The simulations were usually carried out for 1500 queries, and the last 700 values used to compute the averages. This procedure is

reasonable as the queues would have stabilized by then). It is seen from the figure that for a value of load between 0.05 to 0.525, the error in neglecting the read time is less than 12%. The error increases with the load. This aspect is examined further below.

Secondly, what is the effect of file splitting ? Assuming that the number of cassettes required is a geometric random variable, the splitting probability can be obtained for a parameter p as:

$$p_j = (1-p) \quad , j = 1$$

$$p_j = (1-p)p^{j-1}, j = 2, 3, \ldots$$

The mean delay can be easily computed . The results are shown in Fig-3. First Up and Down time distributions U[8,28] and U[6,21], Read time distributed U[14,40] and the independence assumption were used and the value of p was varied between 0.55 and 1.0. The results are shown in Fig-3(a). It can be seen from the figure that the delay increases by 4.8 times with increased splitting compared to the 1.8 times increase in delay for the same load increase when there is no splitting. Also shown alongside are results when the independence assumption was dropped and comparisons made using $\alpha = 0.75$. There is no significant increase in delay due to the dependence as seen from figure. In Fig-3(b) the error due to neglecting the read time is plotted for two cases: Read time distributed as U[14,40] and as U[14,35] using the same Up and Down distributions. In the former case the error is less than 18% and in the latter it drops to less than 9%.

Therefore we can conclude that: (i) The error due to neglecting the read times depends on the overlap between the reading and the combined (convolved) distribution of Up and Down times, and (ii) the error is larger for higher traffic levels. A justification for this approximation can be given now: the performance analysis issues really become interesting when the retrieval times dominate. And if that were not the case the analysis provides reasonable bounds for the delay at moderate traffic levels. If the read times are significant then the appropriate model to use is a tandem queue with finite or infinite buffer space depending on the configuration of the storage system.

Finally, the poisson distribution for arrival distribution of queries can be dropped, and approximations can be given. In the case of mass storage devices, an additional and quite reasonable assumption is that the inter-arrival time distribution is regular, i.e. has small value of the squared coefficient of variation, *scv*. As a justification, the scenario of a large number of users who may retrieve a cassette at any given time with a small probability should suffice. Excellent bounds are available for the mean delay (see Wolff (1990) p.476). For example,

Let c_a^2 and c_g^2 stand for the squared coefficient of variation of the inter-arrival time and service time
Let $\lambda = ET$ and $\rho = \lambda ES$
Then $ED \doteq \frac{(c_a^2+\rho^2 c_g^2)}{2(1-\rho)\lambda}$ (6)

The use of this formula will be in the case the read times cannot be neglected. In that case we model the system as a tandem queue and compute the *scv* of the departure process from the two server queue representing the robots. This can be used to calculate the delay at the second queue using (6). Note that the Uniform distributions for the Up and Down process makes the arrival process rather regular at the read head. So the delay at the read head will mostly depend on the

second moment of the read time and the load at that stage of the tandem system. The additional delay experienced at that stage will almost always be due to reading one cassette and so will be small in magnitude, explaining the small values of error seen so far.

2. The Two Server Case: The next step is to analyze the effect of two robots. When there is only one channel of communication the relevant question to ask is what improvement can be obtained by replacing the two robots by a single one that is twice as fast? Before examining this issue, we note that increasing the number of robots without sacrificing efficiency is beneficial. But the standard multi-server approximations do not apply here as the robots more often than not will work together on servicing a request. Due to this parallelism, closed form expressions for the delay are not easily obtained. Instead we take the route of bounding the performance via the fast server model. In this process in this and the next section, some related issues related to control of the robots are examined.

Proposition I: When the number of cassettes required is only one per query the single fast robot outperforms the two robots in the convex sense ($\leq_{cx}$) in terms of delay in the system experienced by a query, if D_1 and U_1 are either identically distributed or have the same functional form of a unimodal distribution with $ED_1 < EU_1$; and no advance information on cassette location for a query is passed on to the storage system prior to completion of service of all preceding queries.

Proof: Let the service time for the two robot case be simply U_1 . This is a lower bound in the stochastic sense (i.e. $\leq_{st}$ compared to the actual service time experienced (see Stoyan (1983) for definition)). The service time for the fast single robot is $(D_1+U_1)/2$. This service time is stochastically smaller in the convex sense than U_1 under the assumptions of the proposition. The

result now follows from the recursion for the delay (2). That is in the modified two robot system with U_1 as the service time:

$$W_1 = U_1$$
$$W_2 = [W_1 - T_1]^+ + U_2 = \max[U_2, U_1 + U_2 - T_1]$$
$$W_3 = [W_2 - T_2]^+ + U_3 = \max[U_3, U_3 + U_2 - T_2, U_3 + U_2 + U_1 - T_1 - T_2] \quad \text{etc.}$$

We can permit advance information to be passed on, but if in addition add the assumptions that (i) queries have to be served in *FCFS* order ; and (ii) a robot will get *blocked* (i.e cannot do further work) if the previously cassette has not been loaded onto the read head then,

Proposition II: Under the same assumptions as in proposition I, but allowing for advance information to be passed on to the storage system, but in the presence of blocking, we obtain the that the delay in the fast server is stochastically smaller ($\leq_{st}$) than in the two server system.

Proof: The blocking leads to a cyclic assignment of queries to the two robots, with the first robot handling the 1st, 3rd, 5th,... and the second robot handling the 2nd, 4th, ... query. In addition there is conflict between the robots which leads to:

$$W_1 = U_1$$

$$W_2 = \max\,[\,[U_1 - T_1]^+ ,\, U_2]$$

$$W_3 = \max[\,\max\,[\,0,\, U_1 + D_1 - T_1 - T_2\,] + U_3,\ \max\,[\,0,\ U_2 - T_2\,]\,]$$

And in general $W_{2n+1} > \max\,[\,\max\,[\,0,\, U_{2n-1}+D_{2n-1}-T_{2n-1}-T_{2n},\, U_{2n-1}+D_{2n-1}-T_{2n-1}-T_{2n} + U_{2n-3}+D_{2n-3}-T_{2n-3}-T_{2n-2}, \ldots\,] + U_{2n+1},$

$$\max\,[\,0,\, U_{2n}-T_{2n},\, U_{2n} + U_{2n-2}+D_{2n-2}-T_{2n}-T_{2n-1}-T_{2n-2}, \ldots\,]\,]$$

$$\geq \max[\,0,\, U_{2n}/2-T_{2n}/2,\, U_{2n}/2 + (U_{2n-1}+D_{2n-1})/2 + (U_{2n-2}+D_{2n-2})/2 - T_{2n}-T_{2n-1}, \ldots\,] + U_{2n+1}$$
$$\geq_{st} \max\,[\,0,\ (U_{2n+1}+D_{2n})/2 - T_{2n},\ (U_{2n+1}+D_{2n})/2 + (U_{2n-1}+D_{2n-2})/2 - T_{2n}-T_{2n-1}, \ldots\,] + U_{2n+1}/2$$

which proves the result when compared to (2) for the fast single server.

What happens when more than one cassette has to be retrieved per query in the presence of blocking? Here the blocking is under the assumption that cassettes must be read in exactly the order specified by the external system, and a robot gets blocked if the previously requested cassette has not been read. The result is identical as once again a cyclic assignment takes place. So we can state,

Proposition III: Under the same assumptions as in proposition II, even when a query may require more than one cassette to be retrieved the delay in the case of the fast single robot is stochastically smaller.

Proof: Identical to proposition II, where on each sample path the terms $U_i + D_i$ are replaced by the terms corresponding to the number of cassettes requested and retrieved by each of the two robots.

Simulation results are shown in Fig-4, that compare the fast robot with the two robot system operating under the assumption that cassettes will be read only in the sequence requested but that there will be no blocking, which is better than the blocking delay. In Fig-4(a) the distributions of the Up, Down and Read times were U[8,16], U[6,12] and U[12,28]. There is a 1.9% increase in mean delay due to not reading in sequence . In Fig-4(b) the three distributions were set to U[4,28], U[3,21] and U[8,40]. In this case the penalty increases to 3%.

This leads us to consider the possibility of letting the cassettes be read in the *sequence retrieved* rather than in the *sequence requested.* So the *FCFS* assumption is dropped as far as reading is concerned. But retain the assumption that requests for cassettes are still passed on to the storage system in the order of the queries. This makes the two robot system similar to a two server queue operating under the FCFS queue discipline ,i.e., the first free robot serves the next request for a cassette.

Certainly the situation is now better in the two robot case as the cyclic assignment is known to be inferior (see Wolff, p.502). Note that in this proposed scheme we should even allow two requests to be *intermingled.* That is, the two robots attend together to a query, but if a robot falls idle it attends to the next request, and in that case overtaking is allowed. Thus the cassettes are read in the sequence retrieved, and the information so obtained is managed by the user (external system).

Proposition IV: Let $s > max(U_1 + D_1)$. This is the maximum time to manipulate a cassette. Let $W_n^{(j)}$ for $j = 1,2$ stand for the delay experienced by the *nth request* under the fast robot and two robot system. Then under the assumptions in the last paragraph,

$$W_n^{(2)} \leq_{st} W_n^{(1)} + 2.5s$$

Proof: Throughout assume that the two robots work together on a query by sharing the work. The superscripts (1) and (2) will be used to denote which system is being analyzed. If a robot

(say #1) goes idle and the other robot (#2) is still busy then: (i) there is no work for robot 1 under the no idling assumption, and robot 2 is doing a single *fetch* (Down plus Up) operation and has no other work to do at the moment. The assumption is that the service time for fetching is bounded by s. In making the comparisons assume that the same sequence of up and down times are used to compare the behavior of the two systems. Then by (i) and (ii) the extra delay till service commences for the *nth* query , $D_n^{(2)} - D_n^{(1)}$, is bounded by $1.5s$. This can be inferred from a sample path construction by looking at the last instant either of the two robot was idle and assuming that the fast robot was free at that instant. Then the $1.5s$ bound follows by the fact that the extra work to be done by a robot in system (2) is utmost one Up and Down plus one Down time larger before commencing service on a new request when compared to the fast robot system. The extra service time to do the fetching for the *nth* request is utmost s larger for system (2) because it may not be possible to allocate the work evenly between the two robots. The proposition follows.

Remarks: (i) This result shows that if several cassettes are to be retrieved on a regular basis then the overhead in terms of possible extra delay for the two robot system is small. This may be a small penalty to pay when compared to additional benefits from improved reliability. This trade-off can be analyzed using simulation or analytical methods.

(ii) Under the assumption of proposition IV, when there is only one cassette to be fetched per query, the penalty in terms of additional delay due to the 2 robot configuration could be large under light loads. This statement can be explained when the fetch times are deterministic; otherwise simulation is the best method for verifying this conjecture. An example is provided in

Fig-5. The Up and Down distributions used were U[8,16] and U[6,12]. In the first case considered, $p_1 = 1$. The penalty is seen to be 77-96%. Next the split used is $p_1 = 0.5$ and $p_2 = 0.5$ and the penalty reduces to around 40%. In the last case, the split is into one to six cassettes with probabilities {0.16,0.32,0.48,0.64,0.80,1.00}. The penalty drops dramatically to 8%, confirming the conjecture.

(iii) It is now possible to conclude that when there are few cassettes to be retrieved per query and the reliability of the system is extremely high, the fast robot configuration is better. In all other cases, a suitable retrieval strategy coupled with multiple robots will provide better overall performance. The strategy should allow adequate flexibility in retrieval sequencing as evidenced by the propositions above.

(iv) It is very important to avoid file splitting. As a rule of thumb for the cases considered, the extra penalty paid for file splitting is quadratic in nature. That is say the average number of cassettes required is x, then the increase in delay due to this compared to that due to increasing the traffic load but avoiding file splitting is proportional to x^2.

(v) From the simulation results, a good approximation for the delay in the case of 2 robots and uniformly distributed fetch times is:

$$EW^{(2)} \sim EW^{(1)} + (EU_1 + ED_1)/2$$

3. Two Read Heads/ Two Robots: A curious phenomenon occurs when there are two read heads. When only one cassette is required per query it is clear that the additional read head will be beneficial in reducing the delay. What happens when several cassettes are required to be retrieved per query? Should the robots attend to only one or both or which query? If the queries are sent on FCFS basis to the system, but the storage system is given the flexibility to choose between two queries received, then it appears that an optimal strategy would be to serve only one query at a time. So the second read head serves as a pre-emption mechanism. So what is the optimal policy?

The objective is to minimize the mean delay for serving queries. The assumptions are: (i) Queries will be received in FCFS fashion by the storage system. (ii) But the storage system will have the flexibility of choosing which query to serve. (iii) No idling will be permitted. (iv) Exact service times are known. (v) Once an order has been given to fetch it must be completed before executing any other order by a particular robot. (vi) The Up and Down times are deterministic and given by u and d. The last assumption is made to simplify an otherwise difficult problem.

Proposition VI: Under the above assumptions the optimal policy is to serve the query with the smaller remaining after completion of any pending orders.

Proof: Consider a finite horizon dynamic program where the horizon n is the number of queries to be served. When $n = 2$, assume work is being done on a query (#1) and a second query (#2) arrives. Let the work left to be done by the two robots for completing any pending fetch orders is $r1$ and $r2$. Let the remaining work to be done be $n(1,1)$ u's and $n(1,2)$ d's. By assumption (v),

$n(1,1) = n(1,2) = n(1)$. Let the work required to be done for query #2 be $n(2)\ (u+d)$. Ignore the distinction between the two queries, and write the total work left as $n(1)+n(2)+r1+r2$. It is evident that the optimal policy is to first serve the query with the smaller $n(i)$, $i=1,2$ after completing the remaining fetches (per assumption (v)).

Let the proposition be true when there are m stages to go. Consider the first query once again. Let a second query arrive when the first query is still being serviced. Retaining the previous notation, let $n(1) > n(2)$. Compare the strategy of servicing query #1 first with the following alternate strategy. Serve query #2 first, then keep serving query #1 till a *total* of $n(1)$ fetches have been completed, all the while respecting the *no idling* condition. Then revert to the optimal strategy for m stages. Note that this action leads to the exact final work condition as the strategy of serving query #1 first, it outperforms the action of serving query #1 first; and is sub-optimal by the induction hypothesis. The case $n(2) > n(1)$ can be similarly handled using the *switch* (interchange) argument. The proposition follows.

Remark: When the fetch times are not deterministic the interchange argument fails, but it appears reasonable to conjecture that the same policy will be optimal.

References:

Lavenberg,S.S., *Computer Performance Modelling Handbook*, Academic Press, New York (1983).

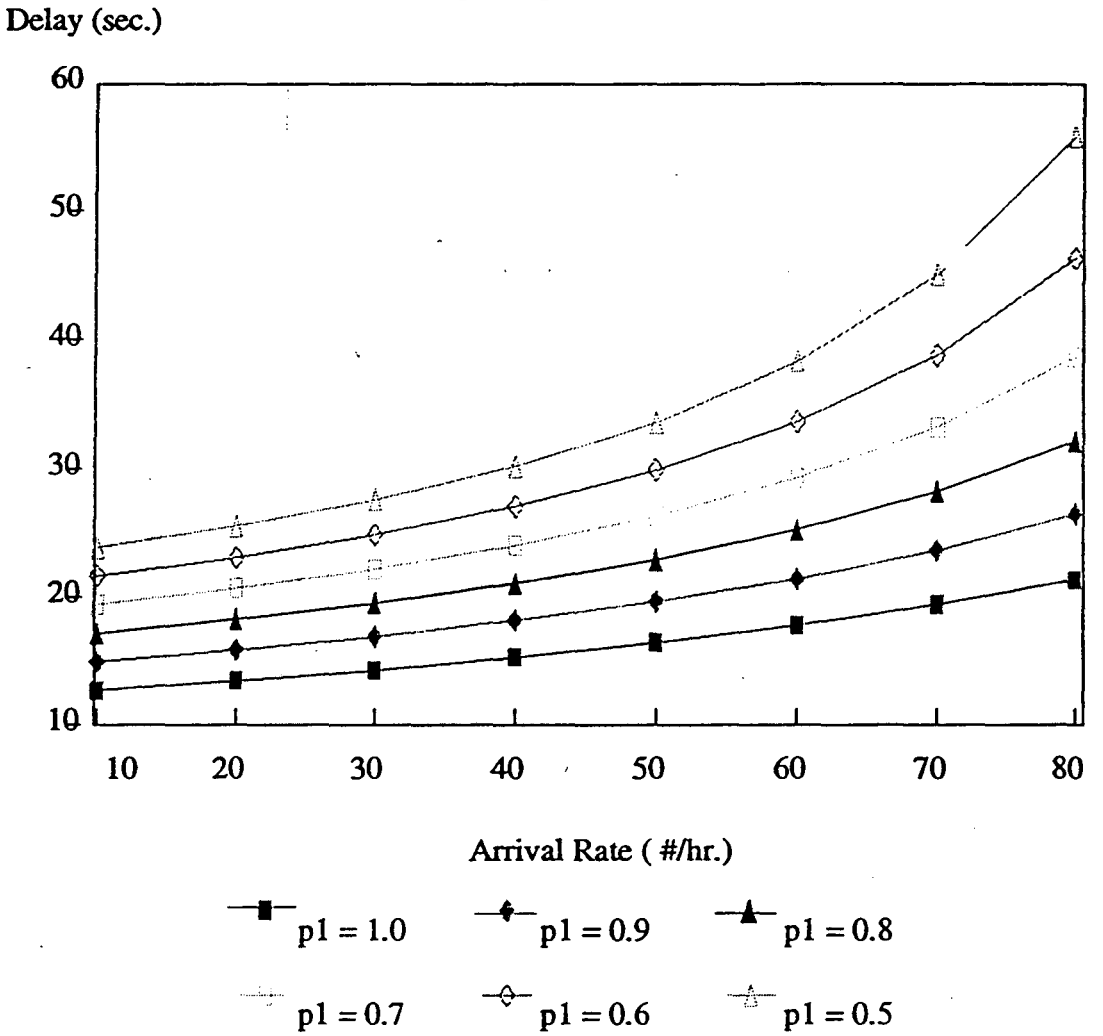
Ranade,S., *Jukebox and Robotic Libraries*, Meckler, Westport, London, (1992).

Stoyan,D., *Comparison Methods for Queues and Other Stochastic Models*, John Wiley & Sons, New York, (1983).

Wolff,R.W., *Stochastic Modelling and the Theory of Queues*, Prentice-Hall, Englewood Cliffs, New Jersey (1990).

Figure 1: M/G/1 Model

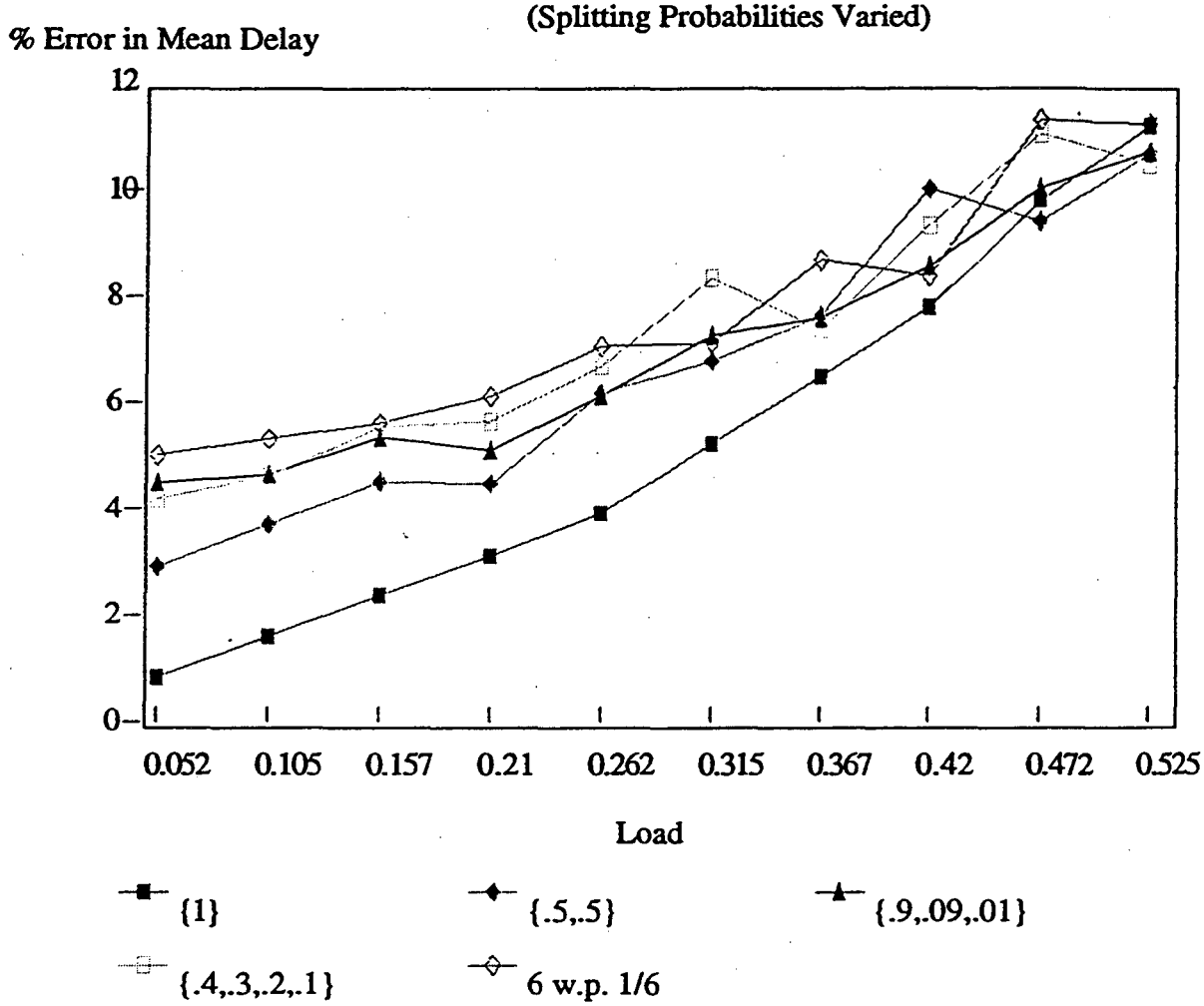
Effect of Splitting Files into 1 or 2 Cassettes



p1 = probability one cassette needed

1 - p1 = probability two cassettes needed

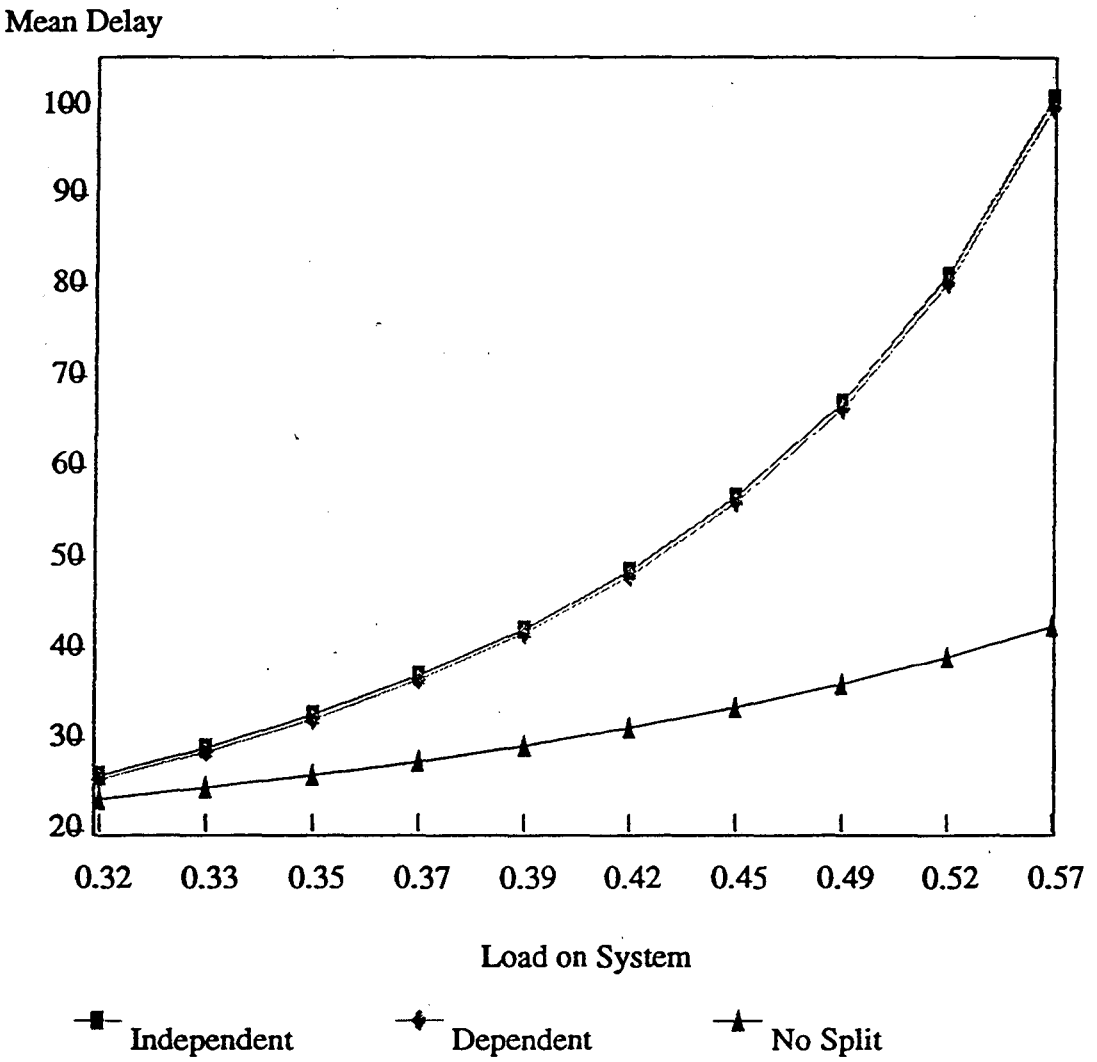
Figure 2: Error Due to Neglecting Read Time
(Splitting Probabilities Varied)



Note: Up ~ U[8,16], Down ~ U[6,12], and Read ~ U[12,28]

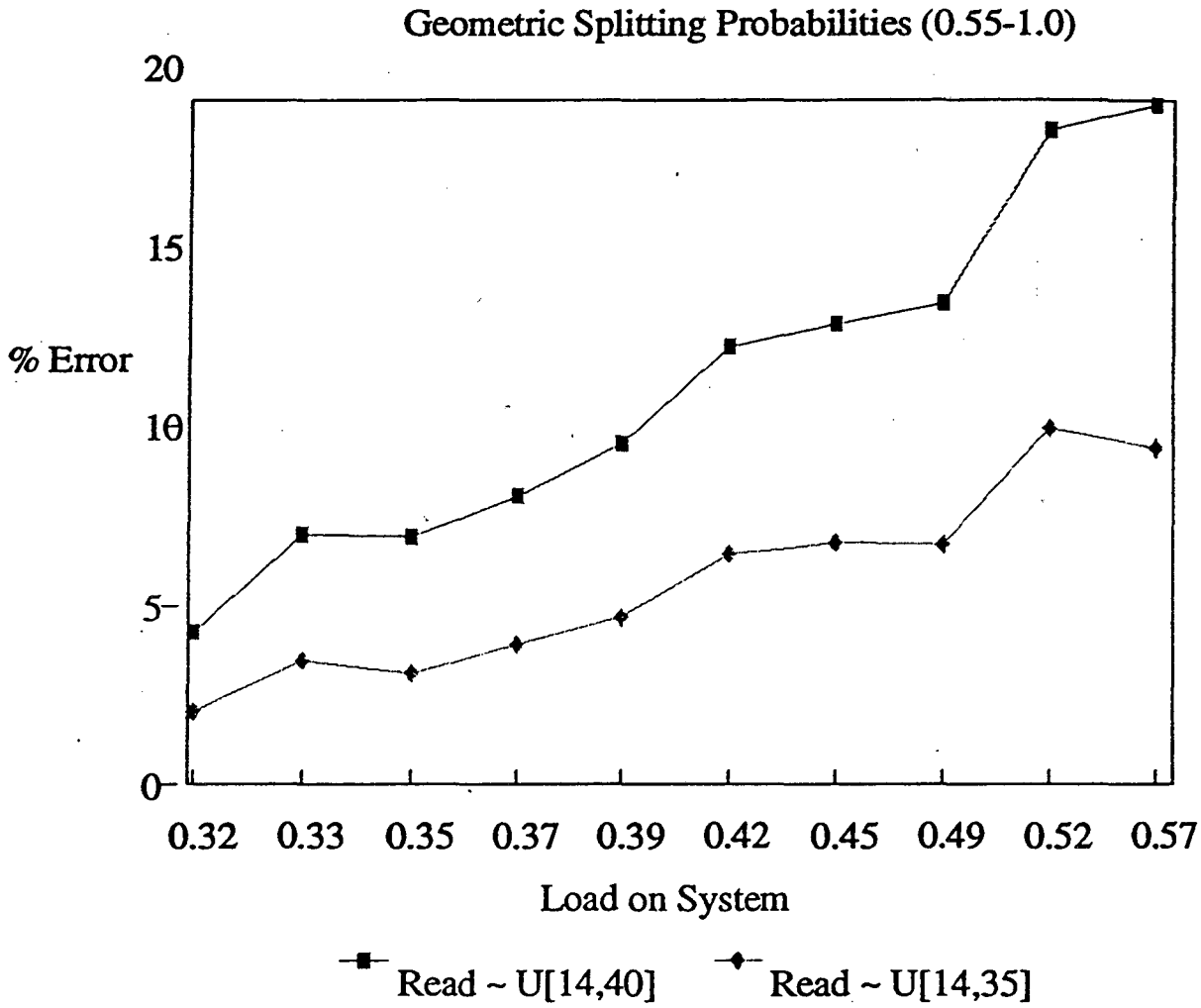
Figure 3(a): Effect of File Splitting on Mean Delay

Geometric Splitting Probabilities (0.55-1.0)



U ~ U[8,28], Down ~ U[6,21], Read ~ U[14,40]

Figure 3(b): Error in Mean Delay due to Neglecting Read Time



U ~ U[8,28], Down ~ U[6,21]

Figure 4(a): Effect of Reading Sequence

Up~U[8,16], Down~U[6,12], Read~U[12,28]

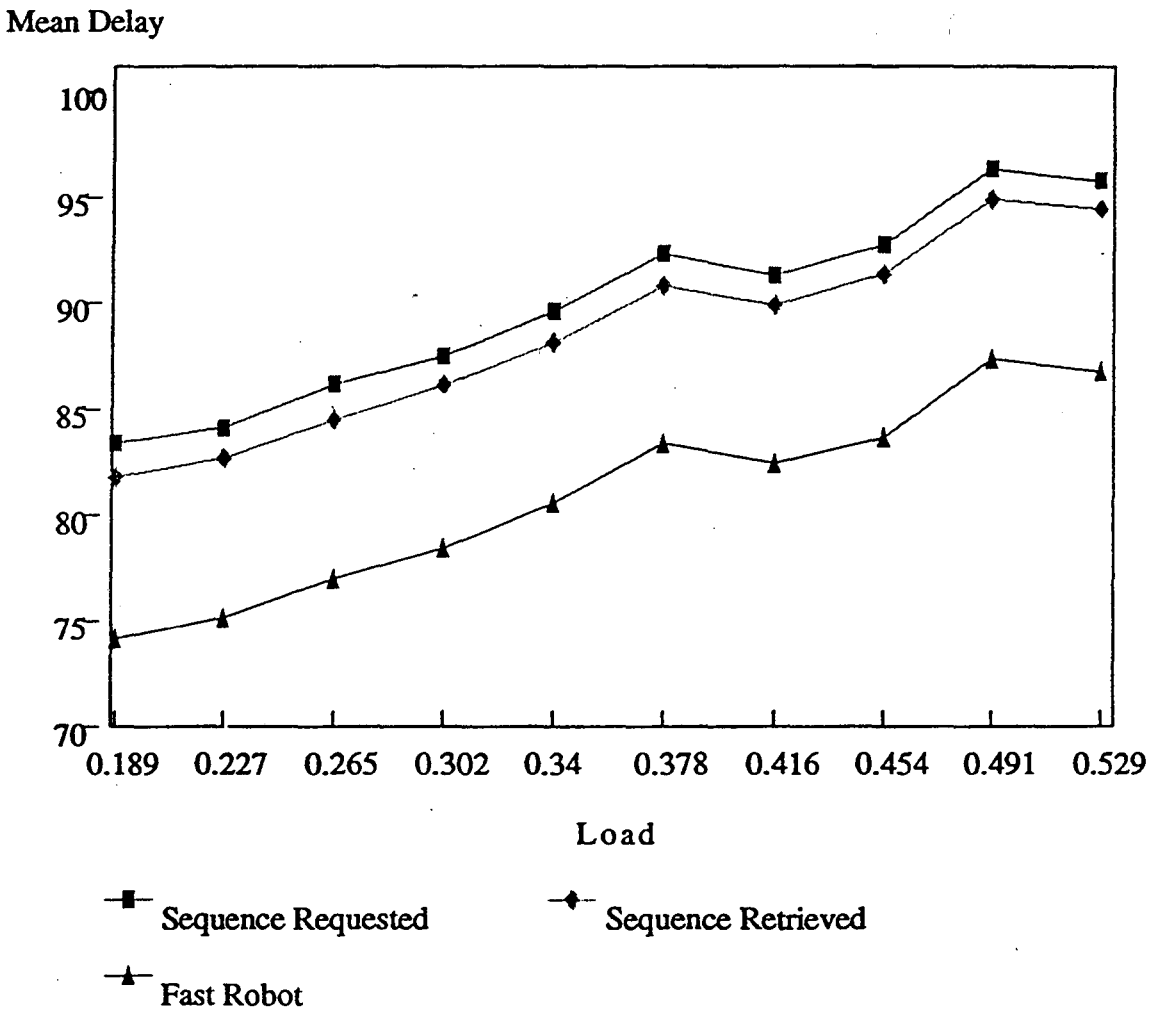


Figure 4(b): Effect of Reading Sequence

Up~U[4,28], Down~U[3,21], Read~U[8,40]

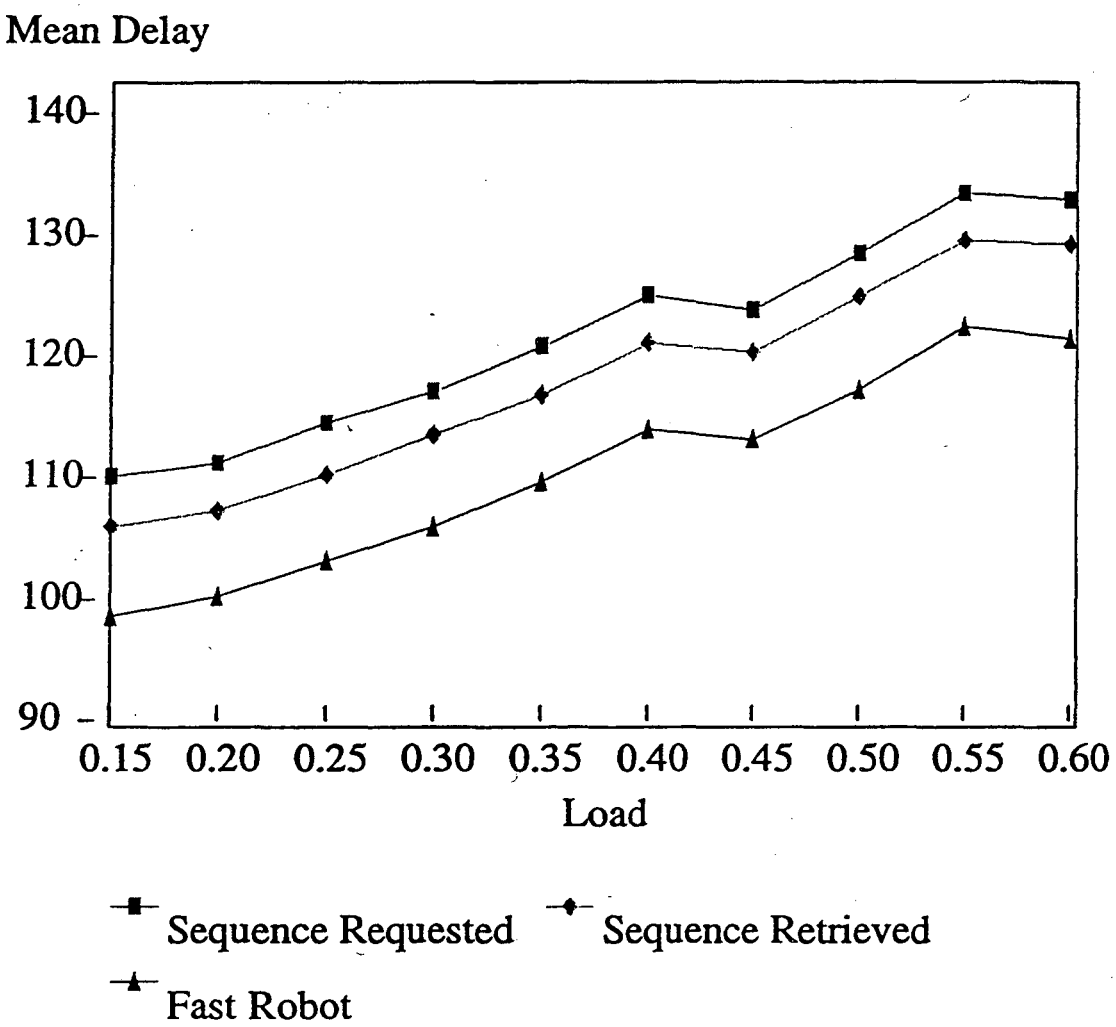
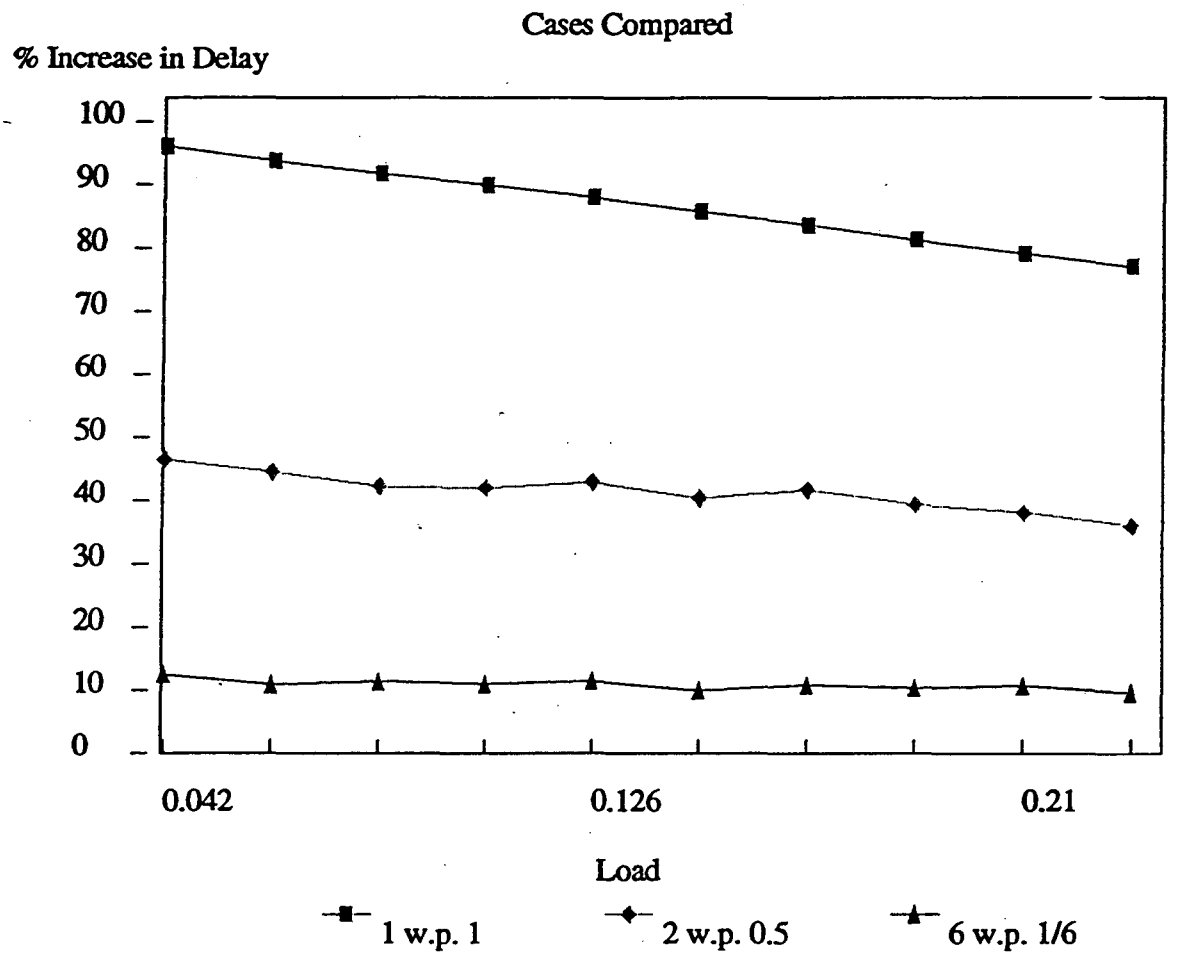


Figure 5: Two Robots or One ?



Up ~ U[8,16], Down ~ U[6,12]

LAWRENCE BERKELEY LABORATORY
UNIVERSITY OF CALIFORNIA
TECHNICAL INFORMATION DEPARTMENT
BERKELEY, CALIFORNIA 94720