The J. of MIS
Research
Vol.2, No.2, Dec 1992

분산된 저장 구조를 지닌 화일에서의 화일 활용성 산정

김 성 언*

Estimation of the Utilization of a File with Distributed Free Space

Analysis for the file with distributed free space such as a VSAM has been done. Birth and death process has been adopted to describe the status changes of control intervals in a VSAM file. File utilization is calculated as a function of the number of control intervals which contain different number of records in them. Effect of the control interval sizes and the loading factors are analyzed over the time horizon.

Ⅰ. Introduction

The research effort addresses the file usage changes through the modeling procedure of physical design for growing databases in the environment of the distributed free space such as in a virtual storage access method(VSAM) file. A VSAM grows out of the need for an access method that allows data to be accessed both directly by key and sequentially in key-defined collating order[Keehn and Lacy, 1974]. The way a VSAM file reads a record is similar to that of an indexed sequential access method(ISAM) file except that the operation is not described in terms of tracks and cylinders. In a VSAM file, instead of cylinders that are subdivided into tracks, control areas are subdivided into control intervals[Martin, 1975].

A control interval(CI) is a continuous area

*우신투자자문(주)

of a direct access storage that a VSAM file uses to store data records and to control information that describes these records. A CI is the unit of information that a VSAM file transfers between virtual storage and disk storage. Whenever a record is retrieved from a direct access storage, the entire CI containing the record is read into a VSAM file I/O buffer in virtual storage. The CIs in a VSAM file data set are grouped together into fixed-length contiguous areas of direct access storage called control areas(CAs). The number of CIs for each CA is fixed. The maximum size of a CA is one cylinder, and the minimum size is one track of a Direct Access Storage Device storage.

In a VSAM file, unused space can be scattered throughout the data set as free space. The space that was occupied by the deleted record is available as free space, because when a record is deleted, the record is physically erased. Insertions into a key-sequenced data set use the free space provided during the definition of the data set, provided by the record deletions, or developed as a result of CI and CA splits. Unlike a hashed file and an ISAM file, a VSAM file does not have an overflow area. Instead, it handles the overflow problem through CI splits and CA splits. With the increase of records in a database, CIs are filled up. If there is not enough free space in a CI to accommodate the new records, then a CI split takes place at the point of insertion. When a CI split occurs, the records comprising the first half of the full CI are copied into the new CI and the old CI is adjusted by shifting the latter half of the records to the front. The index is updated to include the new CI at the appropriate point. The last record provides the key value for the index entry[Wiederhold, 1987]. When free space is no longer available in a CA, a split occurs in a manner similar to a CI split. With the growth of the database, the number of CI split increases. Also, with the increase of transactions on the database, the physical structure of the file and the file utilization are changed. In this paper, the number of CIs that contains different number of records is estimated using a stochastic model and based on these CIs, file utilization is estimated.

In section 2, main modeling procedure for the physical database design is described. In section 3, numerical experiment is given based on the model in section 2. Conclusion is given in the final section.

Ⅱ. Analysis of the Physical Database

This section describes the modeling of the physical database design for a VSAM file. A VSAM supports three different data set formats : key-sequenced, entry-sequenced, and relative record. In this research, a key-sequenced data set(KSDS) is used as the underlying file structure. A KSDS consists of two physical components on the direct access storage device. The first of these is the index component, which contains the key fields and pointers to the location of the record to which that key field belongs. The second is the data component, which contains the records that hold the user data including the key field. The key field is small compared to the whole record; similarly, the index component is small compared to the data component. Therefore, the main concern of this research is the data component. In a KSDS, logical records are placed in the data set in ascending collating sequence by key field. Records can be retrieved and inserted, both randomly and sequentially. When a KSDS is created, unused space, called free space, can be scattered throughout the data set to allow records to be inserted. Free space within a CI is used for in-place reorganization of KSDS's for additions, updates, or deletions of records within that CI[IBM Corp., 1985]. To some extent, this helps to keep the data components in physical sequence in spite of many subsequent random insertions. However, as a result of excessive random insertions and deletions, the components eventually become physically out of sequence, although they are still in logical sequence when accessed through the index of the cluster[Ranade and Ranade, 1986].

The main purposes of this section are to develop a model that describes such a physical database deterioration by analyzing the expected behavior of a physical database structure in which splits are used to handle overflows, and to calculate the file utilization using the number of CIs which will be obtained through the model. Similar works related to the overflow handling and the utilization calculation have been done in the environment of 2-3 trees and B-trees[Chu and Knott, 1989; Eisenbarth, 1982; Quitzow-and Klopprogge, 1980; Yao, 1978]. However, most researches are applied for a static situation without considering the time variations, and no research is applied for a KSDS in a VSAM file.

A problem of handling the model comes from the fact that the probabilities of inserting or deleting a record into or from each CI continuously changes with time. If the model deals with the logical deletion instead of the physical deletion as have been practiced by many researchers for the ISAM file analysis, the modeling procedure is a lot easier. Another problem in modeling comes from a CI split. Since the split changes the state of a CI, it is difficult to define any distribution function that describes the number of records in each CI, differently from the analytical method used in an ISAM file analysis.

1. Assumptions

Basically, it is assumed that no record spans over CI boundary are allowed, and that the data record size is fixed. The major assumptions given in this modeling are as follows. First, records are uniformly distributed according to keys of them over the smallest and the largest keys. Second, insertion is at random. If there are n keys in a database, these n keys divide all possible key values into n+1 slots. If an insertion has an equal probability of being in any one of these n+1 slots, it is called a random insertion and its probability is given by $1/(n+1)$. Third, the probability of a record being deleted is assumed to be equal for all records, i.e., $1/n$. Fourth, the arrival of records to be inserted to the physical database follows a Poisson process with arrival rate λ and each record in the database and in the arriving stream has a unique key. The transitions between CIs can be used to model the insertion process. When a record arrives for an insertion into the database, it is assigned to a CI. When a record is inserted into a database, its service is assumed to have commenced. The service continues until a record is physically deleted from the system, and the space occupied by the record is then immediately reusable by another record. Finally, it is assumed that the service times of all records inserted into the file are independently, identically distributed random variables with an exponential distribution and a finite mean $1/\mu$. The rate that an insertion occurs between time t and $t+\Delta t$ is equal to $\lambda \Delta t$, and the rate that a deletion occurs between this time interval is equal to $\mu \Delta t$.

2. Modeling

A major source of difficulty in modeling arises from the need to keep track of the number of CI's. CI^i is defined as a CI which contains i records. This number is influenced by the rate of arrival of new records into each CI, by the deletion rate of records, and by the way of handling overflows; i.e., CI splits. These in turn depend on the probabilities of a record insertion and deletion into and from each CI. The birth and death process has been adopted as a means of describing the behavior of the expected number of CI's at time t.

Let $X_i(t)$ be the number of CI^is. It is assumed that the probability that a record is inserted into a CI^i is $iX_i(t)/(n+1)$, and the probability that a record is deleted from a CI^i is $iX_i(t)/n$ at time t, where n is the total number of records in the system at time t and $n=n(t)$. There are two occasions where $X_i(t)$ can increase. One occurrence is an insertion into the CI^{i-1}s, and the other is a deletion from CI^{i+1}s. $X_i(t)$ decreases when an insertion or a deletion occurs from the CI^is. In other cases, $X_i(t)$ remains the same. This transaction follows a birth and death process. The instantaneous changes in the system state can amount to an increase(birth) or a decrease (death) of a CI^i. When a transaction occurs, the following conditional transition probabilities are given using a state of $X_i(t+\Delta t)$ for a given state $X_i(t)$, where $i=2, 3, \cdots, b-1$ and b is the maximum number of records that a CI can hold as explained before.

$$\Pr\{X_i(t+\Delta t)=a2 \mid X_i(t)=a1, X_{i-1}(t), X_{i+1}(t)\}$$

$$=[\frac{(i-1)X_{i-1}(t)}{n+1}\lambda_n \Delta t] + [\frac{(i+1)X_{i+1}(t)}{n}\mu_n \Delta t] \qquad (1)$$

for $a2=a1+1$

$$=[\frac{iX_i(t)}{n+1}\lambda_n \Delta t]+[\frac{iX_i(t)}{n}\mu_n \Delta t] \qquad (2)$$

for $a2=a1-1$

$$=1-(1)-(2), \qquad (3)$$

for $a2=a1$

where λ_n and μ_n are the insertion rate and the deletion rate, and a1 and a2 are constant, which indicate initial number of CIs at time t and at time $t+\Delta t$ respectively.

The conditional probability equations (1) through (3) can be interpreted in the following ways. The first term of the equation (1) represents the birth of a record into one of the CI^{i-1}s, with a probability $(i-1)X_{i-1}(t)\lambda_n\Delta t/(n+1)$. The second term represents the death at one of the CI^{i+1}s, with a probability $(i+1)X_{i+1}(t)\mu_n\Delta t/n$. The result of these combinations is an increase in the number of CI's. The interpretation of equation (2) is similar to that of equation (1). This equation represents the birth or the death of a record at one of the CI's with a probability $iX_i(t)\lambda_n\Delta t/(n+1)$ or $iX_i(t)\mu_n\Delta t/n$, respectively. The result is a decrease in the number of CI's. In other cases that do not appear in equation (1) and (2), the number of CI's remains the same as shown in equation (3). These equations also can be applied for B-trees analysis[Quitzow and [illegible], 1980].

Let b be the capacity of a CI and k be $(b+1)/2$ and assume that b is an odd number for the simplicity of the analysis.

When $i=1$, $i=b$, or $i=k$, the equation (1) is [illegible] in the following way.

i) When $i=1$

$$\Pr\{X_1(t+\Delta t)=a2 \mid X_1(t)=a1, X_2(t)\} = [\frac{2X_2(t)}{n}\mu_n\Delta t] \qquad (4)$$

The interpretation of equation (4) is that the increase of X1(t) is caused only by the deletion on one of the CI2s with a probability $2X_2(t)\mu_n\Delta t/n$.

ii) When $i=b$

$$\Pr\{X_b(t+\Delta t)=a2 \mid X_b(t)=a1, X_{b-1}(t)\} = [\frac{(b-1)X_{b-1}(t)}{n+1}\lambda_n\Delta t] \qquad (5)$$

The interpretation of equation (5) is that the increase of $X_b(t)$ is caused only by the insertion on one of the CI^{b-1}s with a probability $(b-1)X_{b-1}(t)\lambda_n\Delta t/(n+1)$.

iii) When $i=k(=(b+1)/2)$, one more equation that reflects the CI split is added to equations (1), (2), and (3).

$$\Pr\{X_i(t+\Delta t)=a2 \mid X_i(t) = a1, X_{i-1}(t), X_{i+1}(t), X_b(t)\} = [\frac{bX_b(t)}{n+1}\lambda_n\Delta t] \qquad (6)$$

for $a2=a1+2$

The interpretation of equation (6) is that the increase of $X_k(t)$ has an additional source besides the sources in equation (1). The birth of a record into one of the CIbs, with a probability $bX_{b-1}(t)\lambda_n\Delta t/(n+1)$, causes a CI split and, with this split, $X_k(t)$ is increased.

Using the conditional probabilities (1) through (6), first, the expected values of $X_i(t+\Delta t)$, conditional on $X_i(t)$, $X_{i-1}(t)$ and $X_{i+1}(t)$, can be obtained, second, the unconditional expectation for $X_i(t+\Delta t)$ can be obtained by applying another expectation on the expected values of $X_i(t+\Delta t)$. That is, $E[E[X_i(t+\Delta t) \mid X_{i-1}(t), X_i(t), X_{i+1}(t)]]$ becomes $E[X_i(t+$

$\Delta t)$][Ross, 1984]. Therefore, the following equation is obtained.

$$E[E[X_i(t+\Delta t) \mid X_{i-1}(t), X_i(t), X_{i+1}(t)]]$$
$$=E[X_i(t+\Delta t)]$$
$$=\frac{(i-1)\lambda_n\Delta t}{n+1}E[X_{i-1}(t)]$$
$$+\frac{(i+1)\mu_n\Delta t}{n}E[X_{i+1}(t)]$$
$$-\frac{i\lambda_n\Delta t}{n+1}E[X_i(t)]-\frac{i\mu_n\Delta t}{n}E[X_i(t)]$$
$$+E[X_i(t)] \qquad (7)$$

After we subtract $E[X_i(t)]$ from both sides of equation (7), if we divide them by Δt and take the limit on Δt, we will have the following differential difference equation:

$$\lim_{\Delta t\to 0}\frac{E[X_i(t+\Delta t)]-E[X_i(t)]}{\Delta t}=\frac{dE[X_i(t)]}{dt}$$
$$=\frac{(i-1)\lambda_n}{n+1}E[X_{i-1}(t)]$$
$$+\frac{(i+1)\mu_n}{n}E[X_{i+1}(t)]$$
$$-(\frac{i\lambda_n}{n+1}+\frac{1\mu_n}{n})E[X_i(t)] \qquad (8)$$

Substitute $E[X_i(t)]$ with $Y_i(t)$ and let λ_n and μ_n be λ and μ. Then, equation (8) can be rewritten as

$$Y_i'(t) =\alpha(i-1)Y_{i-1}(t)+\mu(i+1)Y_{i+1}(t)$$
$$-(\alpha+\mu)\,i\,Y_i(t), \qquad (8')$$

where $Y_i'(t)=dY_i(t)/dt$, $\alpha=\lambda/(n+1)$, and $i=2, 3, \cdots, b-1, \neq k$. This equation is similar to the equation that Quitzow and Klopprogge [1980] derived in terms of pages considering both insertions and deletions for the B−trees model.

When $i=1$ or $i=b$, we can derive equation (9) or (10) using a procedure similar to that used in equation (7) through (8) to derive equation (8′).

$$Y_1'(t)=2\mu Y_2(t)-(\alpha+\mu)Y_1(t) \qquad (9)$$

$$Y_b'(t)=\alpha(b-1)Y_{b-1}(t)$$
$$-(\alpha+\mu)\,b\,Y_b(t) \qquad (10)$$

If an insertion occurs on one of the CI^bs, then a CI^b splits. When this happens, half the records remain in the split CI, and the other half move to a new CI within the same CA or to that in a different CA, depending on space availability. As a result of the split of one of the CI^bs, two CI^ks are generated. Therefore, when $i=k$, this CI split effect is added to the differential equation in the following way:

$$Y_k'(t)=\alpha(k-1)Y_{k-1}(t)+\mu(k+1)Y_{k+1}(t)$$
$$-(\alpha+\mu)\,k\,Y_k(t)+2\alpha\, b\, Y_b(t). \qquad (11)$$

The solution for the system of differential difference equations given in equations (8′) through (11) can be obtained by an initial condition method. Consider a database that was initially created with n_0 fixed-length records. Let $F(0)$ denote the total number of CIs assigned when ξ number of records are initially loaded into a CI, where ξ is called the loading factor. Since each CI has ξ records on file creation, it allows that $F(0)$ equals to $[n_0/\xi]$. The initial conditions(i.c.) that reflect this fact can be set up by

$$Y_i(0)=F(0) \qquad \text{for } i=\xi$$
$$Y_i(0)=0 \qquad \text{otherwise.} \qquad (12)$$

Equations (8′) through (11) can be represented as a normalized system of b first-order, linear equations. These equations form an initial condition problem with given initial conditions in equation (12).

This initial condition problem can be represented in a vector form:

$$Y'(t) = A(t)\, Y(t), \tag{13}$$

i.c.

$$Y(0) = [0\ 0 \cdots F(0) \cdots 0\ 0]^T,$$

where

$$Y(t) = [Y_1(t)\ Y_2(t) \cdots Y_b(t)]^T$$

and

$$A(t) = \begin{bmatrix} -\beta & 2\mu & & & & 0 \\ \alpha & -2\beta & 3\mu & & & \\ & \cdot & & & & \\ & \cdot & & & & \\ & (i-1)\alpha & -i\beta & (i+1)\mu & & \\ & \cdot & & & & \\ & \cdot & & & & \\ & & (k-1)\alpha & -k\beta & (k+1)\mu & 2\alpha b \\ & & & \cdot & & \\ & & & \cdot & & \\ 0 & & & \cdot & & \\ & & & & (b-1)\alpha & -b\beta \end{bmatrix} \begin{matrix} \text{row} \\ 1 \\ 2 \\ \cdot \\ \cdot \\ i \\ \cdot \\ \cdot \\ k \\ \cdot \\ \cdot \\ \cdot \\ b \end{matrix}$$

where $\alpha = \lambda/(n+1)$, $\beta = \alpha + \mu$ and $n = n(t)$.

The solution procedure of equation (13) requires the total number of records in the database. The fact that n is the function of time makes getting the solution difficult. n(t) can be obtained by solving the following ordinary differential equation.

$$\frac{dn}{dt} = \lambda - \mu n \tag{14}$$

Equation (14) indicates that insertion increases, and deletion decreases, the number of records in the system. The general solution of this equation is given by

$$n = \lambda/\mu + C \exp(-\mu t), \tag{15}$$

where C is an arbitrary constant that should be determined by the initial condition. As the database was created with n_0 records initially, we can decide C using the initial condition; i. e., when $t=0$, $C = n_0 - \lambda/\mu$ Using this constant C, we can determine the particular solution of equation (14).

$$n = \lambda/\mu + (n_0 - \lambda/\mu) \exp(-\mu t) \tag{16}$$

The system of differential equation (13) could not be solved in an analytical method, because n is a function of time and so is the A matrix. Numerical method is used to obtain the solution of this equation, and the simulation package SLAM Ⅱ has been used for this purpose. SLAM Ⅱ uses Runge-Kutta-Fehlberg(RKF) algorithms to integrate the equations[Pritsker, 1986]. The RKF method is particularly useful if certain coefficients in the differential equation are empirical functions for which analytical expressions are not known, and hence for which initial series developments are not feasible. This method provides the specific capability of numerically integrating a system of first-order ordinary differential equations such as equation (13). Using this package, the time history of Y(t) for a given equation for Y'(t) is obtained. This is accomplished by considering Y(t) as a function of the derivatives of Y(t) using a

Taylor series expansion. The values of Y(t) can be estimated by evaluating the Y′(t). Let $F_i(t)$ be the solution of equation (13), then the minimum number of CIs required to accommodate the records is given by

$$CI_{min}=\sum_{i=1}^{b} i\, F_i(t)/b. \qquad (17)$$

Using CI_{min}, utility of the file, U(t) can be calculated. Utility represents that portion of the file that is utilized or being used efficiently. Therefore, maximum value of U(t) can be obtained as follow:

$$U(t)=\sum_{i=1}^{b} i\, F_i(t)/(b\sum_{i=1}^{b}F_i(t)) \qquad (18)$$

Equation (18) shows that at a given time, utilization of a file depends on the CI capacity and the total number of CIs.

Ⅲ. Numerical Experiment

The previous section focusses on developing a model that describes the behavior of a physical database in a VSAM file. The model considers that the database is built up through both insertions and deletions of records.

Consider a database that has 50,000 records initially. This size of this database file is small to medium by conventional commercial standards. The two key parameters that determine the evolution of the physical database system from state to state are the record arrival rate to the database and the deletion rate of records from the database. In the development of the model, the arrival rate per period has been assumed to be distributed as a Poisson random variable, and the deletion time distribution per record has been assumed to be exponentially distributed. For the numerical experiments, it is assumed that the arrival rate of records to be inserted into the database is 200 records per period. The unit of a period is assumed to be one hour. Each record is assumed to be deleted at a rate of 0.001 per period. This means that if there are N_0 records in the system, the rate at which records are deleted is 0.001 N_0, and this rate is changed as the number of records in the system is changed over time. The choice of the values of the insertion rate and deletion rate combination is based on the assumption that the database grows over the time horizon. Besides the two parameters mentioned above, two other parameters are considered: CI capacity sizes, and the range of loading factors for a given CI capacity size. In performing the numerical analysis, a finite set of parameter values is considered to be tested.

The growth of a database over time from a given initial state is illustrated in Figure 1 for different CI capacity values when $\lambda=200$, $\mu=0.001$, and $N_0=50{,}000$. To find the effect of the CI capacity sizes, three different sizes are used in the experiment: 9, 15, and 21. Numbers here indicate the maximum number of records per CI and these numbers can be changed to any numbers from 3 to 21, for example, for the proper comparison considering both the size of a CI and the moderate size of each record. To set up the same initial condition in the analysis, the number of records to be loaded initially are arranged as 6, 12, and 14, respectively, yielding same utility value of 0.6667. Under this initial condition, the total initial number of CIs for each database is 8,334, 5,000, and 3,572, respectively. As can be expected, the number of CIs increases as time increases and the number of CIs in the file with small capacity increases faster than the others. However, the relative increases in the total number of CIs, compared to the initial number of CIs, are similar for all files.

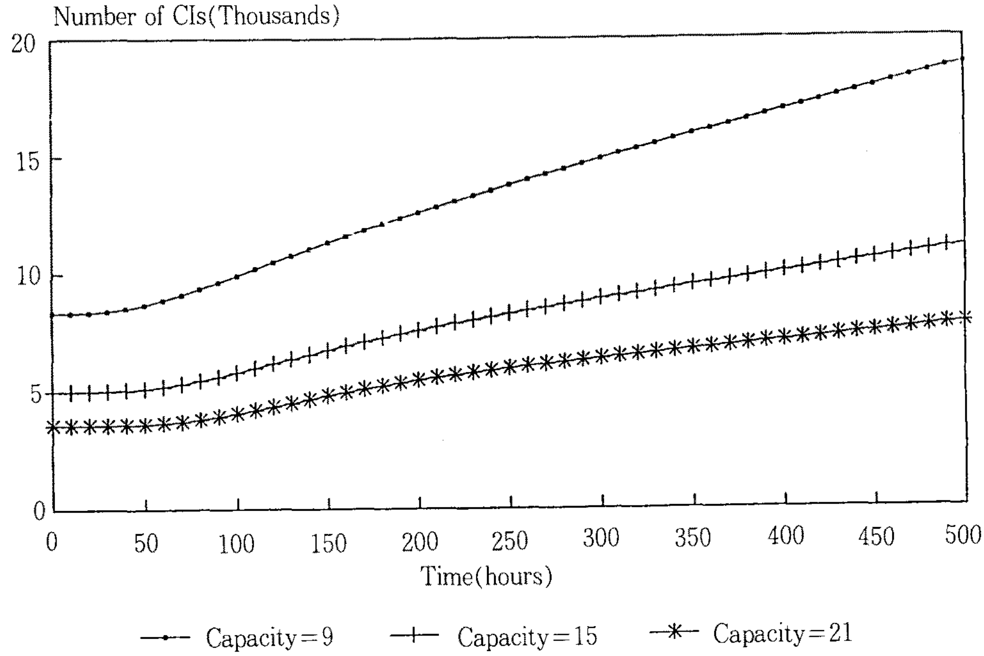

Figure 1. Growth of the CIs for three different CI capacity sizes

The file sizes are doubled at time 400 hours regardless of the CI sizes. Another comparison of the behaviors of files with different CI capacities is shown in Figure 2, which depict the utility changes over the time horizon for each file.

As indicated, the time horizon can be divided into three parts according to the relative pattern of utilities of the files. In the early stage of the file life, utilities increase continuously until they reach their maximum utility values at a time between 60 hours and 70 hours, and decrease thereafter. The greater the CI capacity, the higher the utility values, until the time about 170 hours. At this point, utility values for all files become almost the same with a value of 0.69. After this time, the utility patterns change. Utility is higher for the file that has a smaller CI capacity until the time 300 hours. After the time 300 hours, the file with the greater CI capacity has higher utility values again. What can be obtained from these observations is as follows. First, the utilities are greater than the initial

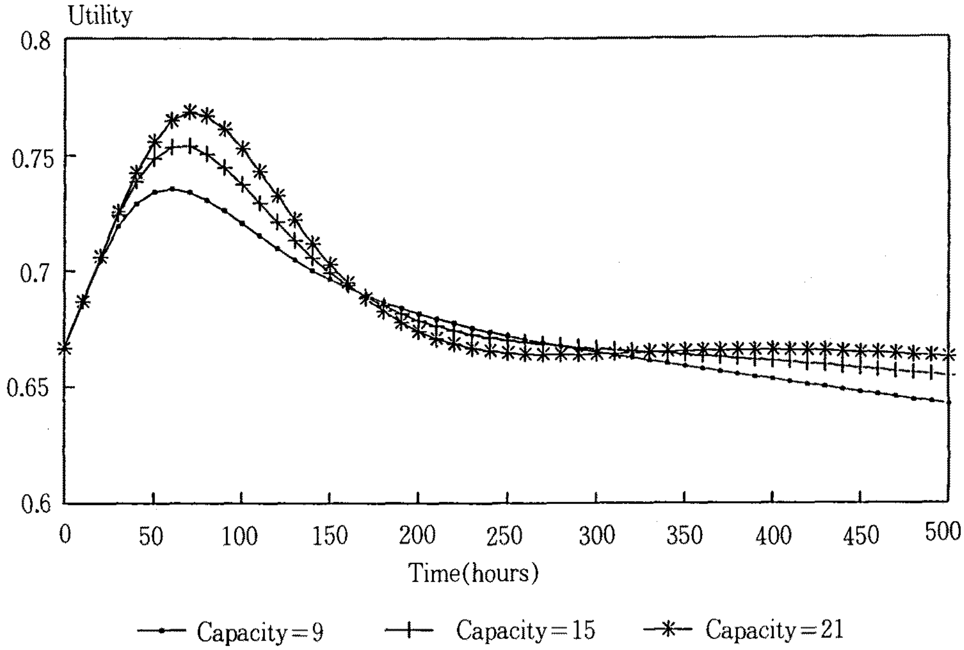

Figure 2. Utility changes for three different CI capacity sizes

utility until the time of 300 hours. Thereafter, they decline to a value less than that of the initial utility; i.e., file utilization deteriorates after the time of 300 hours. Second, if the

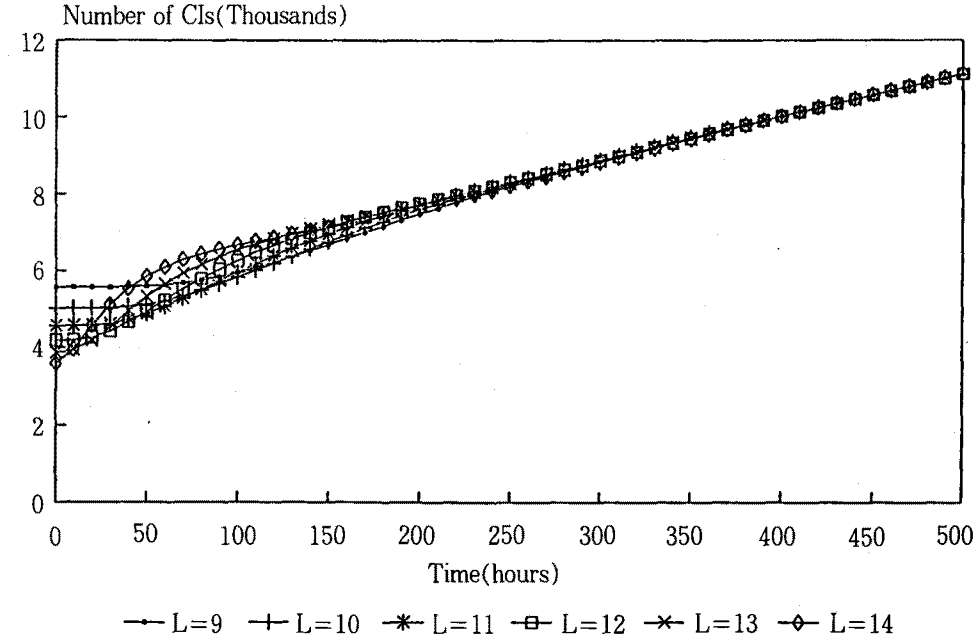

Figure 3. Effect of loading factors on the total number of CIs in a file

utility is a measure of the efficient file use related to the CI size, the optimal CI size will be a function of time. Although the choice of favorable CI size is a function of time, if the length of the time period and the magnitude of the utility differences among the different files of different CI sizes are considered, one would recommend the use of the large CI size. This conclusion can be supported also by a comparison of the total number of CIs in each file as follow. The increase in the total number of CIs during the 500 hours period for the file with b=9 is 10,526 and for the file with b=21 is 4,259. This indicates that the file with b=9 experienced more costly CI splits and occupied more space than the file with b=21. To see how the file behaves for the different loading factors, an experiment was given with initial loading of 9, 10, 11, 12, 13, and 14 records for the file whose CI capacity is 15. It seems evident that, if smaller loading factor is used, greater number of CIs is required. However, this is true only for the file that has a short life time. If the file life is long enough($\geq$400 hours), the total number of CIs becomes almost the same regardless of the loading factor(Figure 3).

A similar situation could be obtained for the utility analysis. At the initial stage of the file, it is difficult to predict the relationship between the utility and the loading factor. However, when the time period becomes long enough, the utilities of all cases converge to one value. The utilities become about 0.67 at time 300 hours and about 0.65 at time 500 hours for all cases(Figure 4).

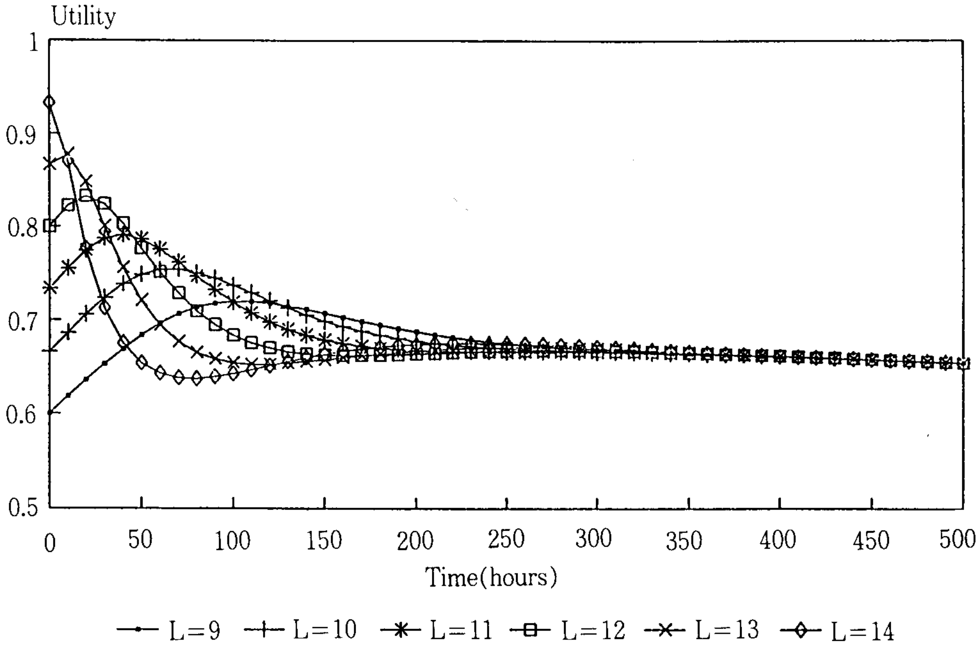

Figure 4. Effect of loading factors on the utility of a file

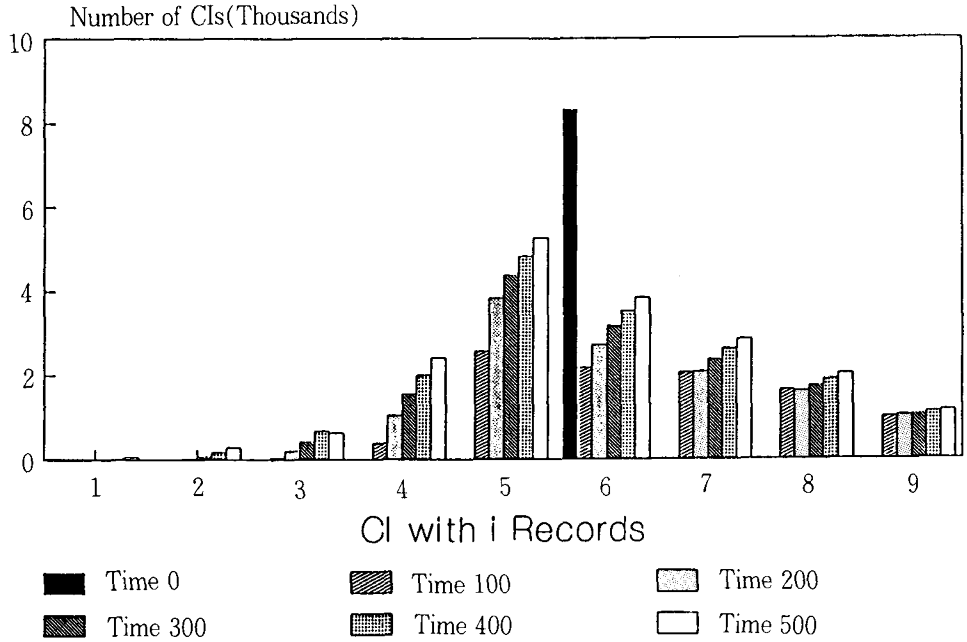

Figure 5. CI spreading pattern over the entire CI's after initial loading

These results suggest that loading factor does not influence the file status much if the file has a long life time.

The distribution change of the number of CIs over the CI's was examined for the file whose CI capacity is 9. This experiment shows how the file structure changes over time for the given parameter values. At the file creation time, records are loaded with a loading factor 6; so, $X_6(0)$ is 8334 and $X_i(0)$ $(1 \le i \le 9,\ i \ne 6)$ are zero. After the loading time, the number of CIs is spread over the other CI's as a result of records insertions and deletions and CI splits. Figure 5 shows that, at time 100, $X_5(100)$ is even bigger than $X_6(100)$ and retains the bigger values over the entire time horizon. The reason for this is that $X_5(t)$ has an extra increasing source from the CI splits. When the time period becomes big enough, $X_i(t)$ $(1 \le i \le 4)$ begin to grow significantly and, before time 400, $X_4(t)$ becomes even bigger than $X_8(t)$ and $X_9(t)$. This situation indicates that file utilization decreases with the increase in the number of CIs that contains a smaller number of records. Both the increase in the number of total CIs and the decrease in utility remains as the evidence of file usage deterioration with increasing transactions on the database.

Ⅳ. Conclusion

The structure of the VSAM file under records insertion and deletion is changing dynamically with time. Birth and death process made it possible to describe the dynamical status change of the CIs.

Experiment results of the VSAM file shows the followings : First, the greater the CI capacity, the higher the utility value. Second, the optimal CI size is the function of time if it is determined based on the utilization of a file. Third, loading factors do not have influence on the utility much. Finally, if the file life is long enough, the file utility is deteriorated. The last fact suggests a need for a mechanism, such as a file reorganization, to preserve a file in the good usage condition.

REFERENCES

Chu, J.H, and Knott, G.D., "An Analysis of B-Trees and Their Variants", *Information Systems*, 14, 1989, pp.359-370.

Eisenbarth, B., "The Theory of Fringe Analysis and Its Application to 2-3 Trees and B-Trees", *Information and Control*, 55, 1982, pp.125-174.

IBM Corp., *MVS/Extended Architecture VSAM Administration Guide*, San Jose, California, (1985).

Keehn, D.G. and Lacy, J.O., "VSAM data set design parameters", *IBM Sys.* J., 3, 1974, pp. 186-202.

Martin, J., *Computer Data-Base Organization*, Prentice Hall, 1977.

Pritsker, A.A.B., *Introduction to simulation and SLAM II*, Halsted Press Book, John Wiley & Sons, New York, 1986.

Quitzow, K.H. and Klopprogge M.R., "Space Utilization and Access Path Length in B-trees", *Information Systems*, 5, 1980, pp.7-16.

Ranade, J. and Ranade H., *VSAM: Concepts, Programming, and Design*, Macmillan Publishing Company, New York, 1986.

Ross, S., *A First Course in Probability*, Macmillan Publishing Company, New York, 1984.

Wiederhold, G., *File Organization for Database Design*, McGraw-Hill, New York, 1987.

Yao, A.C., "On Random 2-3 Trees", *Acta Informatica*, 9, 1978, pp.159-170.

◇ 저자소개 ◇

저자 김성언은 서울 대학교 자연과학대를 졸업하고, 미국 Louisiana State University에서 Computer Science 석사와 경영학 박사 학위를 취득하였다. 현재, 국민 대학교, 건국 대학교 및 성신 여자 대학교에서 강의를 하고 있으며, 우신 투자 자문㈜에서 자문위원으로 재직하고 있다. 경영정보학을 전공하였으며 주요 연구 관심은 데이타베이스 분야에 있다.