Optimal Backup Policies for a Database System with Periodic Incremental Backup

Cunhua Qian¹, Yingyan Huang¹, Xufeng Zhao¹,², and Toshio Nakagawa²
¹Nanjing University of Technology, College of Management Science and Engineering, Nanjing, China
Email: qch64317@njut.edu.cn
²Aichi Institute of Technology, Department of Marketing and Information System, Toyota, Japan
Email: toshi-nakagawa@aitech.ac.jp

Abstract—This paper considers the following backup scheme for a database system: a database is updated at a nonhomogeneous Poisson process and an amount of updated files accumulates additively. To ensure the safety of data, full backup are performed at time NT=L or when the database fails, whichever occurs first, and between them, incremental backups are made at periodic times iT (i=1, 2, ..., N-1) so as to make the backup efficiently. Using the theory of cumulative processes, the expected cost is obtained, and an optimal numbers N* of incremental backup which minimizes it given T or L are analytically discussed. Finally, it is shown as an example that optimal numbers are numerically computed.

Index Terms—database, incremental backup, cumulative damage model

I. INTRODUCTION

In recent years, database in computer systems have become very important in the highly information-oriented society. The database is frequently updated by adding or deleting data files, and is stored in floppy disks or other secondary media. Even high reliable computers might sometimes break down eventually by several errors due to noises, human errors and hardware faults. It would be possible to replace hardware and software when they fail, but it would be impossible to do a database. One of important things for using computers is to backup data files regularly, i.e., to copy all files in a secondary medium. Fukumoto et al. discussed optimal checkpoint generations for a database recovery mechanism[1].

Cumulative damage models in reliability theory, where a system suffers damage due to shocks and fails when the total amount of damage exceeds a failure level K, generate a cumulative process [2]. Some aspects of damage models from reliability viewpoints were discussed by Esary et al. [3]. It is of great interest that a system is replaced before failure as preventive maintenance. The replacement policies where a system is replaced before failure at time T [4], at shock N [5], or at damage Z [6,7] were considered. Nakagawa and Kijima [8] applied the periodic replacement with minimal repair [9] at failure to a cumulative damage model and obtained optimal values T*, N* and Z* which minimize the expected cost. Satow et al. [10] applied the cumulative damage model to garbage collection policies for a database system. Qian et al. [11-13] successfully obtained the optimal Full, Incremental and Cumulative Backup policies for a database by using cumulative damage models.

In this paper, we apply the cumulative damage model to the backup of files for database media failures, by putting shock by update and damage by updated files: a database is updated at a nonhomogeneous Poisson process and an amount of updated files accumulates additively. To lessen the overhead of backup processing, incremental backups with small overhead are adopted between full backups. The mean time to full backup and the expected costs are derived, using the theory of cumulative processes. Further, an optimal number of incremental backup which minimizes the expected cost is analytically derived. Numerical example is finally shown.

II. FULL BACKUP AND INCREMENTAL BACKUP

Backup frequencies of a database would usually depend on the factors such as its size and availability, and sometimes frequency in use and criticality of data. The simplest and most indispensable method to ensure the safety of data would be always to shut down a database, and to make the backup copies of all data, log and control files in other places, and to take them out immediately when some date in the original secondary media are corrupted. This is called the total backup. But, such backup has to be made while a database is off-line and unavailable to its users, and would take more hours and costs as data become larger.

To overcome these disadvantages, export backup has been developed because only a small percentage changes in most applications between successive backups [11,14]: Export backup makes the copies of only files which have changed or are new since a prior backup. The resources required for such backup are proportional to the transactional activities which have taken place in a database, and not to its size. This can shorten backup times and can decrease the required resources, and would be more useful for larger databases. On the other hand, the type of backups, in which the copies of all files are made in a storage area and the attributes of archives while backing up only modified files are updated, is called the full backup. Export backup cannot take the place of full backup; however, it can reduce the frequency of full backup which is required. For most failures, a database can recover from these points by log files and restore a consistent state by the last full backup and export backups.
Incremental backup has been well-known as one of export backups; it makes all copies of modified files since the last full backup or incremental backup and we can restore all data by the last full backup and all incremental backups. From the above point of view, we can reduce the frequency of full backup by incremental backups. In this paper, we apply the cumulative damage model to backup policy for a database and derive an optimal full backup interval and incremental backup interval.

Suppose that a database in secondary media fails according to a general distribution \( F(t) \) and full backup is performed when the database fails. Incremental backups are performed at periodic times \( iT (i=1, 2, \cdots, N-1) \), and export only updated files which have changed or are new since the last full backup or incremental backup. In order to enhance reliability of the series system of incremental backups, full backup is performed at times \( NT=L \).

Taking the above considerations into account, we formulate the following stochastic model of the backup policy for a database system: suppose that a database is updated at a nonhomogeneous Poisson process with an intensity function \( \lambda(t) \) and a mean-value function \( R(t) \), i.e., \( R(t)=\int_0^t \lambda(u)du \). Then, the probability that the \( j \)-th update occurs \( (j=0,1,\cdots) \) exactly during \((u,v)\) is

\[
H_j(u,v) = \frac{[R(v)-R(u)]^j}{j!} e^{-[R(v)-R(u)]}
\]

(1)

where \( R(0)=0 \) and \( R(\infty)=\infty \).

Further, let \( \gamma_j \) denote an amount of files, which changes or is new at the \( j \)-th update. It is assumed that each \( \gamma_j \) has an identical probability distribution \( G(x) = \text{Pr}[\gamma_j \leq x] \) \( (j=1,2,\cdots) \). Then, the total amount of updated files \( Z_j=\sum_{i=j}^{\infty} \gamma_i \) up to the \( j \)-th update where \( Z_0=0 \) has a distribution

\[
\text{Pr}[Z_j \leq x] = G^{(j)}(x) \quad (j=0,1,2,\cdots),
\]

(2)

and \( G^{(j)}=1 \) for \( x \geq 0 \), where \( G^{(j)}(x) \) \( (j=1,2,\cdots) \) is the \( j \)-fold convolution of \( G(x) \) with itself.

Let \( Z(t) \) be the total amount of updated files at time \( t \). Then, the distribution of \( Z(t) \) [3] is

\[
\text{Pr}[Z(t) \leq x] = \sum_{j=0}^{\infty} H_j(t)G^{(j)}(x)
\]

(3)

where \( H_j(t)=H_j(0,t) \).

III. OPTIMAL FULL BACKUP TIME

A. Expected Cost

We discuss optimal full backup interval \( L^*(T)=N^*T \)

Let us introduce the following costs: cost \( c_F \) is suffered for the full backup; cost \( c_D+c_F \) is suffered for the incremental backup when the amount of export files at the backup time is \( x \); cost \( c_{FD}+jc_{FY}+c_F \) is suffered for recovery if database fails when the total amount of import files at the latest incremental backup time is \( x \), where \( j \) is unique integer \( j \leq N-1 \). Further, if the database fails during \([T, (i+1)T]\), then the expected recovery cost till time \( iT \) is

\[
C_{DF}(iT) = \sum_{j=0}^{\infty} H_j(iT)\int_0^\infty [c_D+c_F(x)]dG^{(j)}(x).
\]

(4)

We define time between two full backups as one cycle, then mean time to full backup is

\[
E(L) = NT\bar{F}(NT) + \int_0^{NT} F(t)dt.
\]

(5)

From (4), the total expected backup cost of one cycle is

\[
C_{ED}(N) = C_F + \int (F(iT)-F((i-1)T))\sum_{n=1}^{N-1} C_{ED}(m,T)
\]

(7)

and from (5), the total expected recovery cost of one cycle is

\[
C_{ED}(N) = \sum_{i=1}^{N} (F(iT)-F((i-1)T))C_{ED}(i-1,T).
\]

(8)

Therefore, the expected cost per unit time in the steady-state is

\[
C(N) = \frac{C_{ED}(N)+C_E(N)}{E(L)}.
\]

(9)

B. Optimal Policy

We discuss optimal value \( N^* \) which minimize the expected cost \( C(N) \) in (9). Suppose that the database is updated at a homogeneous Poisson process with an intensity function \( \dot{\lambda}(t)=p\lambda \), and it fails according to distribution \( \bar{F}(t)=1-e^{-\mu t} \), where \( p+q=1 \), further suppose that \( E[Z_j]=1/\mu \) and \( c_F(x)=c_Fx \), then the expected cost in (9) is

\[
\frac{C(N)}{q\lambda} = (c_F-c_{FD}+c_Dp\lambda T)\int_0^{\infty} \frac{c_D+c_{FD}+2c_Fp\lambda T}{1-e^{-q\lambda t}} dt + \breve{C}(N).
\]

(10)

where

\[
\breve{C}(N) = \frac{c_D+c_{FD}+c_Fp\lambda T}{1-e^{-q\lambda NT}} (c_F-c_{FD}+c_Dp\lambda T) Ne^{-q\lambda NT}.
\]

(11)
From (10), we know that optimal \( N^* \) which minimizes the expected cost \( C(N) \) is equality to optimal \( N^* \) which minimizes \( \tilde{C}(N) \).

From the inequality \( \tilde{C}(N+1) - \tilde{C}(N) \geq 0 \), we have

\[
\frac{c_{TD} + c_{F} p \lambda T}{\mu} \geq \frac{c_{F} - c_{D} - c_{F} p \lambda T}{\mu},
\]

where

\[
Q(N) = N - \frac{1 - e^{-\lambda NT}}{1 - e^{-\lambda T}}.
\]

Thus \( Q(1) = 0 \), \( Q(N+1) - Q(N) = 1 - e^{-\lambda NT} > 0 \) and \( \lim_{N \to \infty} Q(N) = \infty \), i.e., \( Q(N) \) is strictly increasing with \( N \).

Therefore, we have the following optimal policy:

1. If \( c_{F} - c_{D} > (c_{TD} + c_{F} p \lambda T)(1 - e^{-\lambda T}) + (c_{F} p \lambda T) \), then there exists a finite and unique minimum \( N^* \) (\( I < N^* < \infty \)) satisfying (12), minimizes \( \tilde{C}(N) \), and

\[
(c_{TD} + c_{F} p \lambda T) \frac{Q(N^*) - N^* e^{-\lambda NT}}{1 - e^{-\lambda NT}} < \tilde{C}(N^*)
\]

2. If \( c_{F} - c_{D} \leq (c_{TD} + c_{F} p \lambda T)(1 - e^{-\lambda T}) + (c_{F} p \lambda T) \), then \( N^* = 1 \), i.e., only full backup needs to be done, and the resulting cost is

\[
\frac{C(1)}{q \lambda} = c_{F} + c_{D},
\]

\( \lambda \) and \( q \), and \( C(N^*)/q \lambda \) decreases with \( \lambda T \), and conversely, increases with \( q \).

IV. OPTIMAL INCREMENTAL BACKUP TIME

A. Expected Cost

It is assumed that a database in secondary media fails according to a general distribution \( F(t) \), and full backup is performed at time \( L \) or when the database fails, whichever occurs first. A database returns to an initial state by full backups. Incremental backups are performed at periodic times \( iT \ (i = 1, 2, \ldots, N-1) \), where \( NT = L \).

In order to ensure point-in-time recovery after media failure, transaction log backups are made since the last full backup, its size depends on the amount of updated data in database. We discuss an optimal incremental backup interval \( T^* \).

Let us introduce the following costs: cost \( c_{F} \) is suffered for the full backup; cost \( c_{D} \) is suffered for one incremental backup or cost suffered for importing by one incremental backup because of exporting and importing data are the reverse process; cost \( c_{D} \) is suffered for every transaction log backup; cost \( c_{F}(x) \) is suffered for reconstructing data by transaction logs when the amount of updated data in database is \( x \), and \( i c_{CD} \) denotes importing cost of incremental backups when the number of incremental backups is \( i \), \( c_{CD} \) denotes importing cost of the last full backup.

Then, the expected cost of incremental backups is

\[
C_{i} = \sum_{j=1}^{i} [F(iT) - F((i-1)T)](i-1)c_{D} + F(L)(N-1)c_{D}, \quad (15)
\]

and the expected cost of transaction log backups is

\[
C_{2} = \sum_{j=1}^{N} \int_{0}^{L} jH_{j}(t) c_{D} F(t) + F(L) \sum_{j=0}^{N} jH_{j}(L)c_{D} \quad (16)
\]

Further, if the database fails during \([i-1), i) \), then the expected recovery cost till time \([i-1) \) is

\[
C_{S} = \sum_{j=1}^{N} [F(iT) - F((i-1)T)](i-1)c_{D}, \quad (17)
\]

and the expected cost of reconstructing data by transaction logs till failure time is

\[
C_{4} = \sum_{j=1}^{N} \sum_{i=1}^{T} \int_{0}^{T} iH_{i}(iT) c_{D} F(t) \int_{0}^{T} jH_{j}(x) dG^{(j)}(x), \quad (18)
\]

Then the total expected backup and recovery cost during the interval \( L \)

\[
E(C) = c_{F} + \sum_{i=1}^{L} C_{i}. \quad (19)
\]

B. Optimal Policy

To discuss optimal values \( T^*(L) = L/N^* \) which minimizes expected cost \( E(C) \) in (19) analytically, we suppose that the database is updated at a homogeneous
Poisson process with an intensity function \( \lambda(t) = p\lambda \), and it fails according to distribution \( F(t) = 1 - e^{-\eta t} \), where \( p + q = 1 \), further suppose that \( E(Y_j) = 1/\mu \) and \( c_x(x) = c_x x \), then the expected cost in (19) is

\[
E(C) = (c_\mu - c_D) + \left( \frac{p}{q} c_L + c_{RD} - c_D \right) (1 - e^{-\eta T}) + p\lambda L c_L + C(N) \tag{20}
\]

where

\[
C(N) = \frac{2c_\mu (1 - e^{-\eta L})}{1 - e^{-\eta N}} - N c_\mu e^{-\eta L} + p N c_\mu (1 - e^{-\eta L}) - \frac{2c_\mu}{\mu} e^{-\eta L} \tag{21}
\]

From (20), we know that optimal \( N^* \) which minimizes the expected cost \( E(C) \) is equality to optimal \( N^* \) which minimizes \( C(N) \). From (21), we know that \( C(\infty) = \lim_{N \to \infty} C(N) = \infty \),

\[
C(I) = (c_\mu + \frac{p c_\mu}{\mu} + \frac{p \lambda L c_L}{\mu}) (1 - e^{-\eta L}) + c_\mu - \frac{p \lambda L c_L}{\mu}
\]

there exists a finite number \( N^* \) \( (I \leq N^* < \infty) \) which minimizes \( C(N) \).

From the inequality \( C(N + I) - C(N) \geq 0 \), we have

\[
2c_\mu (1 - e^{-\eta L}) - \frac{e^{\eta N^*} - e^{\eta L}}{N^* - \eta} - \frac{2c_\mu}{\mu} e^{\eta L} - \frac{p c_\mu}{q} (N + q \lambda L) (e^{\eta N^*} - e^{\eta L}) \geq c_\mu e^{-\eta L} - \frac{p c_\mu}{q} \tag{22}
\]

It is very difficult to discuss the optimal accurate values \( N^* \) accurately. So we seek analytic values \( N^* \) approximately. Optimal accurate numerical values will be given in numerical example and be compared with the approximate numerical values; error analysis will be given at last.

Using two approximations that \([15]\)

\[
e^{-\eta L} \approx 1 - q \lambda L, \quad e^{\eta L} = 1 - \frac{q \lambda L}{N} - \frac{q \lambda L}{N^* + 1}.
\]

thus (21) can be written

\[
\hat{C}(N) = N c_\mu (1 + q \lambda L) + c_\mu \frac{pq(\lambda L)^2}{N\mu} \tag{23}
\]

and the inequality (22) is simplified as

\[
\hat{N}(N + \hat{N}) \geq \frac{pq \lambda L c_\mu}{\mu (1 + q \lambda L) c_D} \tag{24}
\]

Therefore, we have the following optimal policy:

1. If \( \frac{pq \lambda L c_\mu}{\mu (1 + q \lambda L) c_D} < 2 \), there exists a unique optimal \( \hat{N}^* \) \( (1 < \hat{N}^* < \infty) \) satisfies (24) which minimizes \( \hat{C}(N) \), and

\[
2N - 1 < \frac{\hat{C}(\hat{N}^*)}{c_D (1 + q \lambda L)} \leq 2N + 1.
\]

2. If \( \frac{pq \lambda L c_\mu}{\mu (1 + q \lambda L) c_D} \geq 2 \), then \( \hat{N}^* = 1 \), i.e., only backup and log back need to be done, and \( \hat{C}(\hat{N}^*) = \hat{C}(1) \).

C. Numerical Example

Suppose that \( 1/\mu = 1 \). Table II gives the optimal approximate numerical values \( \hat{N}^* \), optimal accurate numerical values \( N^* \) and the resulting costs \( C(N^*) \), \( C(\hat{N}^*) \), \( \hat{C}(N^*) \) and \( \hat{C}(\hat{N}^*) \) for \( \lambda L = 1000, 2000, 3000, 4000 \) when \( c_D = 100 \), \( c_\mu = 10 \) and \( q = 10^{-4} \).

<table>
<thead>
<tr>
<th>( \lambda L )</th>
<th>1000</th>
<th>2000</th>
<th>3000</th>
<th>4000</th>
</tr>
</thead>
<tbody>
<tr>
<td>( N^* )</td>
<td>2</td>
<td>4</td>
<td>7</td>
<td>9</td>
</tr>
<tr>
<td>( \hat{N}^* )</td>
<td>3</td>
<td>6</td>
<td>8</td>
<td>11</td>
</tr>
<tr>
<td>( C(N^*) )</td>
<td>451.075</td>
<td>899.456</td>
<td>1341.765</td>
<td>1776.383</td>
</tr>
<tr>
<td>( \hat{C}(N^*) )</td>
<td>472.086</td>
<td>940.590</td>
<td>1364.307</td>
<td>1818.864</td>
</tr>
<tr>
<td>( \hat{C}(\hat{N}^*) )</td>
<td>719.950</td>
<td>1479.900</td>
<td>2195.586</td>
<td>3037.600</td>
</tr>
<tr>
<td>( \hat{C}(\hat{N}^*) )</td>
<td>663.300</td>
<td>1386.600</td>
<td>2164.888</td>
<td>2994.400</td>
</tr>
</tbody>
</table>

Similarly, Table III gives optimal approximate numerical values \( \hat{N}^* \), optimal accurate numerical values \( N^* \) and the resulting cost \( C(N^*) \) for \( q = 2 \times 10^{-4} \), \( 2.5 \times 10^{-4} \), \( 3 \times 10^{-4} \), \( c_\mu = 5 \), \( 10 \) and \( c_D = 100, 200, 300 \) when \( \lambda L = 4000 \).

These indicate that both optimal numbers \( N^* \) and expected costs \( C(N^*) \) are increasing with \( \lambda L \), \( c_\mu \) and \( q \). Although optimal numbers \( N^* \) are decreasing when \( c_D \) is increasing, expected costs \( C(N^*) \) are increasing.

The reason would be explained that the optimal incremental backup numbers \( N^* \) during time interval \( L \) are related to both costs suffered for incremental backups and costs suffered for reconstructing data by transaction logs: Incremental backup costs are increasing with \( N^* \) and Log Backup costs are decreasing with \( N^* \). So if we want to minimize expected costs \( C(N^*) \), incremental backup numbers \( N^* \) should be decreasing with incremental backup cost \( c_\mu \); on the one hand, costs suffered for reconstructing data by transaction logs are related to \( c_\mu \) and \( q \), when \( c_\mu \) and \( q \) are increasing, reconstructing data costs are increasing if \( N^* \) is constant, so if we want to minimize expected costs \( C(N) \), incremental backup numbers \( N^* \) should be increasing; \( \lambda L \) denotes the frequency of updating of database, if \( \lambda L \) increases, the probability of database failure increases, so reconstructing data costs will increase, optimal
incremental backup numbers $N^*$ should be decreased to minimize the expected costs.

Table II and Table III also give approximate numerical values $N^*$ using (24). Compared with optimal accurate numerical values $N^*$, it is evident that $\tilde{N}^*$ and $N^*$ are not identical. It is interesting that $\tilde{N}^*$ are almost the same with $N^*$ when $q = 2.5 \times 10^{-4}$; when $q < 2.5 \times 10^{-4}$, $\tilde{N}^*$ are greater than or equal to $N^*$, i.e., values of $N^*$ may be $\tilde{N}^* -1$ or $\tilde{N}^* -2$; when $q > 2.5 \times 10^{-4}$, $\tilde{N}^*$ are less than or equal to $N^*$, i.e., values of $N^*$ may be $\tilde{N}^* +1$ or $\tilde{N}^* +2$. So the approximate numbers $N^*$ would be useful for seeking accurate numbers $N^*$.

V. CONCLUSIONS

We have considered two schemes of full and incremental backup for a database system, and have analytically discussed optimal backup policies which minimize the expected cost, using theory of cumulative processes. It would be of interest that there must be optimal number of incremental backups minimizing the expected cost rate. This result would be applied to the backup of a database, by estimating the costs of two backups, recovery, the amount of updated file from actual data and the probability of database failure. However, backup schemes become very important and much complicated, as database systems have been largely used in most computer systems and information technologies have been greatly developed. These formulations and techniques used in this paper would be useful and helpful for analyzing such backup policies.

ACKNOWLEDGMENT

This work is supported in part by National Nature Science Foundation (70801036, 70471017), Humanities and Social Sciences Research Foundation of MOE of China (05JA630027).

REFERENCES


TABLE III.

OPTIMAL NUMBER $N^*$, $\tilde{N}^*$ AND THE RESULTING COST $C(N^*)$, FOR $c_R$, $c_D$ AND $q$

<table>
<thead>
<tr>
<th>$q$</th>
<th>$c_R$</th>
<th>$c_D$</th>
<th>$N^*$</th>
<th>$\tilde{N}^*$</th>
<th>$C(N^*)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>100</td>
<td>9</td>
<td>9</td>
<td>1728.258</td>
<td></td>
</tr>
<tr>
<td>200</td>
<td>6</td>
<td>7</td>
<td>2445.691</td>
<td></td>
<td></td>
</tr>
<tr>
<td>300</td>
<td>5</td>
<td>5</td>
<td>2999.491</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$2 \times 10^4$</td>
<td>100</td>
<td>13</td>
<td>13</td>
<td>2442.371</td>
<td></td>
</tr>
<tr>
<td>100</td>
<td>200</td>
<td>9</td>
<td>9</td>
<td>3456.516</td>
<td></td>
</tr>
<tr>
<td>300</td>
<td>7</td>
<td>8</td>
<td>4234.162</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$2.5 \times 10^4$</td>
<td>100</td>
<td>14</td>
<td>14</td>
<td>2680.863</td>
<td></td>
</tr>
<tr>
<td>120</td>
<td>200</td>
<td>10</td>
<td>10</td>
<td>3792.322</td>
<td></td>
</tr>
<tr>
<td>300</td>
<td>8</td>
<td>8</td>
<td>4645.394</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$3 \times 10^4$</td>
<td>100</td>
<td>11</td>
<td>10</td>
<td>2035.400</td>
<td></td>
</tr>
<tr>
<td>200</td>
<td>8</td>
<td>7</td>
<td>2882.528</td>
<td></td>
<td></td>
</tr>
<tr>
<td>300</td>
<td>6</td>
<td>6</td>
<td>3522.677</td>
<td></td>
<td></td>
</tr>
<tr>
<td>100</td>
<td>19</td>
<td>15</td>
<td>2878.965</td>
<td></td>
<td></td>
</tr>
<tr>
<td>200</td>
<td>11</td>
<td>10</td>
<td>4070.801</td>
<td></td>
<td></td>
</tr>
<tr>
<td>300</td>
<td>9</td>
<td>9</td>
<td>4956.040</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>