動画・音声対応マルチメディア リアルタイムファイルシステム（2）ファイル管理*

2J-7

遠藤幸一郎　森美裕　鈴木希宗子　北川秀雅†
松下電器産業株式会社 情報システム研究所‡

1 はじめに

マルチメディア処理技術の発展と共に、動画(V)、音声(A)を含むマルチメディアデータを統一的に扱う要求が高まっている。[1]

動画、音声はテキストと異なり、時系列に変化するメディアであるため「**リアルタイムの連続入出力が必要**」という制約があり、これが統一的な管理を行うための障害となっている。

ここでは、AVを含むマルチメディアデータの統一的な操作性と、AVデータのリアルタイム連続入出力を満足することを目的とした、離散型のファイル管理方式について述べる。

2 AVファイル管理の問題点

テキストデータ主体の一般的なファイルシステムでは、ファイルを構成するデータは記憶媒体上に離散的に格納されている。これは、編集操作が容易で、記憶媒体の利用効率がよい等テキストデータを扱うには適した方式である。しかしながら、このような離散型のファイルを入出力する場合、個々の離散データへのアクセスがどのようなタイミングで起こるかは予想できない。

AVデータの編集を考えた場合、ビット当たりのコスト等の点から、記憶媒体としては大容量の磁気ディスクや光ディスクが適しているが、これら、アクセス時にシークを必要とする記憶媒体を用いた場合、AVデータのリアルタイム連続入出力を保証するには、シーク時間を補う大容量のバッファが必要になる。

この問題を解決する1手法としては、データを物理的に連続して記録し、入出力途中にシークが入らないようにする方法がある[2]。

しかしながら、この方法では記憶媒体上に連続的なエリアを確保しなければならないため、統一的なファイル管理が困難になるという課題がある。

また、データの連続性を維持したままファイル編集を行うことは困難であり、編集時もしくは定期的に保守を行うことによりデータを連続に並べ直す必要がある。従って、大容量データを扱う場合は、編集の即時性や保守を実施するタイミング等についての課題がある。

データベース管理の1手法として編集の即時性を高め、かつ保守を不要とする方法も提案されているが、データの連続入出力については考慮されていない[3]。

3 ファイル構造

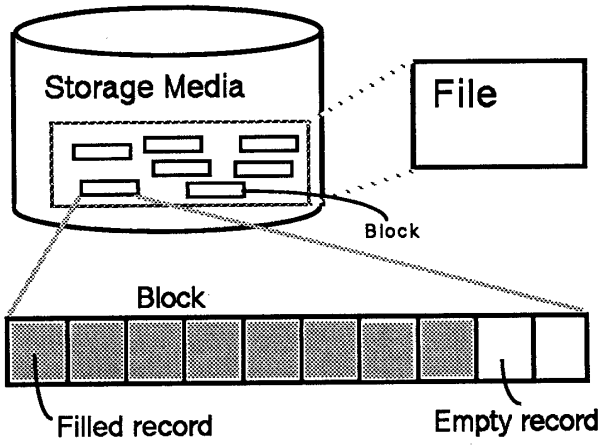

図1: ファイルの物理構造とブロック構造

図1に示すように、本システムのファイル管理部で扱うファイルは、すべて複数のブロックで構成され、各ブロックはディスク上に離散的に置かれている。これは、従来のファイルシステムのファイル構造と基本的に同じである。

ただし、本システムでは、1ブロックのデータを入出力中に、次のブロックへのアクセスが終了するよう、メディアの入出力レートと、記憶媒体の最大アクセス時間、転送時間から、最適なブロックサイズを決定している。従って、少なくともブロックサイズ×2のバッファを持つことで、データ入出力の連続性が保証される。本システムではこの部分をハードウェアで実現している。(詳細はリアルタイムファイルシステム ハードウェア参照)

すべてのファイルをこの単一の構造（離散型）で構成することで、ファイルを扱う部分が共通化され、マルチメディアデータを統一的に管理することが可能となる。

*Multimedia Real-Time File System for Audio and Video (2) File Management

†Koichiro Endo, Yoshihiro Mori, Kisoko Suzuki and Hidemasa Kitagawa

‡Information System Research Laboratory Matsushita Electric Industrial Co.,Ltd.

また、本システムのファイル管理部は、ファイルの編集精度を高めるために、ブロックの中をさらに複数の固定長レコードで論理的に区切っている。ファイルの編集はこのレコード単位で行われる。

レコードには実際にデータが格納されたレコード（有効レコード）と、データの格納されていないレコード（空きレコード）を設けている。ファイル管理部は、ブロック中の有効レコード数が常に一定のレコード数（最低レコード数）以上となるよう管理し、1ブロック中の最低データサイズを保証することでリアルタイムの連続入出力を可能にしている。

最低レコード数は、データの入出力レートと記憶媒体のスペックで決められ、空きレコード数は、編集時に変更するブロックの数（$\propto$ 編集処理に要する時間）と、記憶媒体の有効記憶容量を考慮して決められる。

空きレコードは、高速の編集を実現し、定期的な保守を不要とするために用いる領域である。次にファイル編集について詳細に述べる。

4 ファイル編集

ファイル編集時、レコードの削除もしくは追加によってブロック内の有効レコード数が増減する。ブロック内には空きレコード領域を持っているため、この増減を容易に吸収することができ、編集を高速に処理することが可能になる。しかしながら、編集の状況によっては、ブロック内の有効レコード数が最低レコード数にみたない異常ブロックが発生し得る。

この場合、次のブロックへのアクセス時間に相当するデータサイズが得られないため、リアルタイム連続出力の保証ができなくなる。

本システムのファイル管理部はこの異常ブロックを自動的に検出し、周辺ブロックからのレコード移動を行うことにより、異常ブロックを解消する。このブロック編集処理はファイル編集処理中に実行される。

図2にブロック編集の概念図を示す。

異常ブロックをB_iとし、この異常ブロックに論理的に続くブロックをそれぞれ$B_{i+1},B_{i+2},\cdots,B_{i+j},\cdots$とする。各ブロック中の有効レコード数を$E_0,E_1,E_2,\cdots,E_j,\cdots$とし、これらを加算したものを$S_j=\sum_{k=0}^{j}E_k$で表す。

最低レコード数を$Rmin$、最大レコード数を$Rmax$とすると、すべてのブロック中の有効レコード数を$Rmin$以上でかつ$Rmax$以下にすればよいから、式1をみたす最小のjを求めると、この異常ブロックを補正するに必要な最小ブロック数$j+1$が決まる。

$$N \times Rmin \leq S_j \leq N \times Rmax \ (N\text{は1以上の整数}) \tag{1}$$

$0 < Rmin < Rmax$なる仮定があるので、式1をみたす最小のjを考えた場合、S_jは式2であらわされる。

$$S_j \leq \frac{Rmin \times Rmax}{Rmax - Rmin} \tag{2}$$

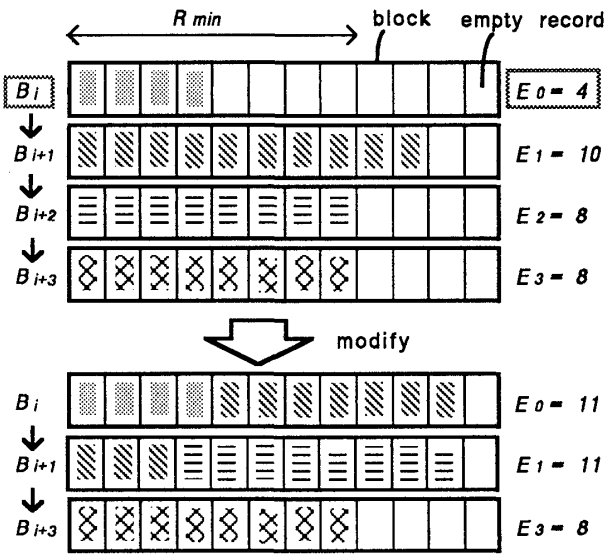

図 2: ブロック編集の概念図

また、$0 < E_0 < Rmin$、$Rmin \leq E_j \leq Rmax$ (j=1,2,$\cdots$)、であるから、

$$j \leq \frac{Rmax}{Rmax - Rmin} \tag{3}$$

となる。以上のように有限個の正常ブロックからレコード移動を行うことにより、異常ブロックを解消できることがわかる。

図2を例にとれば、B_{i+3}以降のブロックは変更しなくともよい。

5 おわりに

以上述べたように、本システムのファイル管理部は、離散型のファイル構造、空きレコードを含むブロック構造、ブロック中データ量の自動管理という特徴を持っている。

本システムを用いることにより、以下のような機能を実現できることがわかった。

1. AVデータのリアルタイム連続入出力
2. 大容量AVデータの即時編集
3. マルチメディアデータの統一的管理

参考文献

[1] 日経エレクトロニクス 特集 マルチメディア始動 No.471(1989).

[2] 富士通 SX／AR カーネル説明書

[3] JAMES MARTIN 著 データベース 日本コンピュータ協会