7B－3　高速UNIXファイルシステムの開発における非同期I/O制御方式の実現

牧 敏行　秋沢 充*1 山下 洋史*2 加藤 寛次*2 鬼頭 昭*3 山田 秀則
日立コンピュータエンジニアリング(株)　*1(株)日立製作所 コンピュータ事業本部
*2(株)日立製作所 中央研究所　*3(株)日立製作所 ソフトウェア開発本部

1．はじめに

近年，UNIXワークステーションのCPU性能は急速に向上しているが，システム性能を引上げるにはファイルアクセス性能の向上が不可欠である。筆者らは，ファイル・ストライピングをベースとした“バーチャルアレイ・ファイルシステム(VAFS)”を提案し，ファイルアクセス性能の向上に取り組んでいる[1]。VAFSのような複数のディスク装置に分割して格納されたファイルに対して高速な入出力を制御するためには，非同期I/Oを実現する必要がある。しかし，現状のUNIXでは標準的な非同期I/O制御方式はない。本稿では，非同期I/O機能をVAFS上で実現する際の課題と解決方法について報告する。

2．VAFS高速化の技術課題

VAFSは1チャネルのSCSIバスに複数のディスク装置(HD)を接続してファイルを分割格納する。アクセスの際には，単一プロセスからの一連の要求を非同期に次々と発行し，並列にHDを動作させてSCSIバスの利用効率を高めることにより，高速なアクセスを実現する機構が必要となる。

従来のUNIXファイルシステム(UFS)には，readの際のディスク先読み(read ahead)とwriteの際のディスク遅延書き込み(delayed write)が用意されている。いずれもカーネル内での部分的な非同期制御であるため，アプリケーションプログラム(AP)レベルでは，前に発行されたアクセス要求の終了を待って次のアクセス要求を発行することになる。また，先読みはシーケンシャルリードに有効であり，ランダムリードでは効果がない。このためシーケンシャルやランダムが混在したAPからの複数HDへのアクセスを多重化させることが困難になる。

この問題を解決するには，APが個々の要求の終了を待たずに次々と発行できるような非同期I/O制御を用意する必要がある。すなわち，前に発行したアクセス要求の終了を待たずに，次のアクセス要求をファイルシステムからデバイスドライバへ発行して，複数のHDを並列動作させることである。

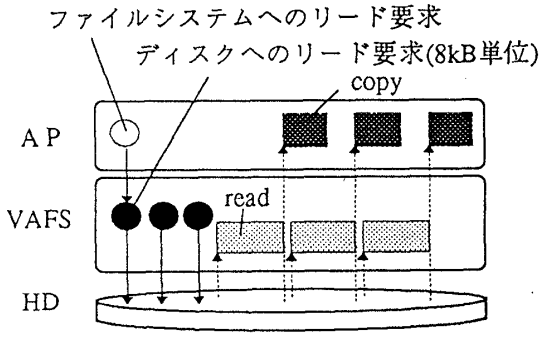

図1　非同期I/O制御のフロー

図1に非同期I/O制御のフローを示す。

非同期I/O制御を行うためには，以下の課題を解決する必要がある。

（1）アクセス要求を発行後，I/O終了を待たずに次のアクセス要求を受け付けるためのアクセス要求発行履歴管理
（2）writeのI/O終了時の後処理およびreadのI/O終了時のシステムバッファからAPバッファへのデータコピーと後処理

この課題を解決するために，非同期I/O要求管理方式とデータコピー制御方式を開発した。さらに，非同期I/Oを行う新規システムコールも開発した。以下これらについて述べる。

3．非同期I/O制御方式

3.1　非同期システムコール仕様

read()およびwrite()システムコールを非同期化したシステムコールとしてasread()とaswrite()を導入する。さらに非同期システムコールを同期化するaswait()システムコールを導入する。aswait()には同期を取る条件として引数で指定する次の3つの機能を実現する。

・WAIT_ALL：すべての非同期I/Oシステムコールが完了するまで待つ。
・WAIT_LRI：少なくとのひとつの非同期I/Oシステムコールが完了するまで待つ。

An Implementation of Asynchronous I/O Control in the Design of Performance Improved UNIX File System
Toshiyuki MAKI, Mitsuru AKIZAWA*1, Hirofumi YAMASHITA*2, Kanji KATO*2, Akira KITO*3, Hidenori YAMADA
Hitachi Computer Engineering Co.,Ltd.
*1 Computer Group,Hitachi,Ltd.
*2 Central Research Laboratory,Hitachi,Ltd.
*3 Software Development Center,Hitachi,Ltd.

表1　システムコールの自動変換

システムコール名		変換方式
read()		asread() + aswait()
write()	delayed	変換せず
	synchro	aswrite() + aswait()

・WAIT_ID：指定した非同期I/Oシステムコールが完了するまで待つ。

また，従来のread()およびwrite()システムコールを使用するAPがVAFSをアクセスする場合には，これらをカーネル内部で非同期システムコールに自動変換する。表1に変換操作の一覧を示す。

3.2　非同期I/O要求管理方式

非同期I/Oシステムコールは，HDへのアクセス要求が未完了のままAPの処理が進むため，I/O完了時に要求発行プロセス自身が後処理を行うことができない。このため，要求の発行履歴をファイルシステムで管理して，I/O完了時にファイルシステムで後処理を行う必要がある。この機能を実現するために，as_request構造体を導入し，システムコール発行単位に履歴を割り当ててリストとして保持する。I/O終了時には本リストを探索して終了処理を行うことにする。本方式により，要求発行元プロセスに対して，I/O終了結果を報告することも可能になる。図2に非同期I/O要求管理方式を示す。

asread()の場合はas_requestのリストにI/O要求を登録した後，HDに読み出し要求を発行する。ここで終了待ち状態に入らず処理を完了し，次のアクセス要求を受け付ける。HDからのI/O終了割込みが発生すると，as_requestのリストから事前に登録したI/O要求をサーチしてシステムバッファからAPバッファへのデータコピーを行う。

一方，aswrite()の場合はAPバッファからシステムバッファへデータコピーを行い，as_requestのリストにI/O要求を登録した後，HDに対してwrite要求を発行する。ここでもasread()の場合と同様，終了待ち状態に入らず処理を完了し，次のアクセス要求を受け付ける。HDからのI/O終了割込みが発生すると，as_requestのリストから事前に登録したI/O要求をサーチしてシステムバッファを解放する。

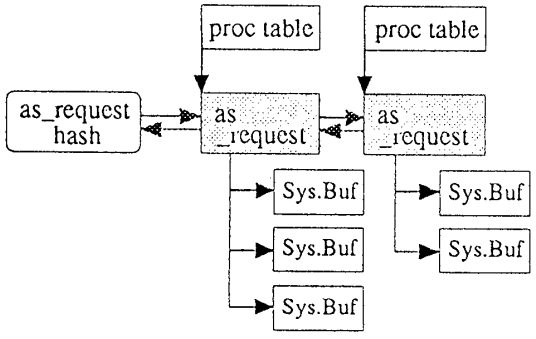

図2　非同期I/O要求管理方式

上記の方式によりファイルシステムで従来の同期制御を非同期化させることを試みたが，このままでは，I/Oサイズがブロックアクセスの単位となる基本ブロックサイズ以下の場合に非同期化されない。これは同一ブロックに対するアクセス要求が重なった場合に，後発の要求は本ブロックが解放されるまで待つことに起因する。そこで後発の要求は発行履歴を残すだけで直ちに発行元に制御を戻し，先発の要求に対するシステムバッファからAPバッファへのデータコピーを行う際に，まとめて後発の要求分もコピーする。

3.3　データコピー制御方式

非同期I/Oによる読み出しでは，発行元のAPは次々とリード要求を発行する。各要求に対するHDからのデータリードが終了した時点ではシステムコールは終了しており，またAPバッファもメモリ上に存在する保証はない。このためreadシステムコールのように個々の要求に対するI/O終了割込み後にシステムバッファからAPバッファへのデータコピーを行う既存の方式では対応できない。そこで，データコピーを行う専用プロセスを用意し，HDからのデータリードが終了した時点では，コピープロセスに通知するのみとし，コピープロセスがスケジュールされた時点でまとめてデータコピーを行う方式とする。

このようにコピープロセスを設けることで，システムコールが終了してもas_requestによるリストを探索して，各APバッファへデータコピーを行うことができる。さらにAPバッファがメモリ上に存在せずにページフォルトが発生してもコピープロセスの延長でページフォルト回復処理を行うことができる。

4．おわりに

VAFS上で稼働する非同期I/O方式を開発した。これにより，単一プロセスで非同期のリード／ライトを実現することができた。また，UNIXの標準インタフェースを介して実現できることを示した。

参考文献

[1]秋沢他5，「バーチャルアレイ・ファイルシステム(vafs)の基本構想」，情処全国大会講演論文集4-61，平成4年10月

注)UNIXオペレーティングシステムはUNIX System Laboratories，Inc.が開発し，ライセンスしています。