YuihaFS:ファイル毎にバージョンを作成できるファイルシステムの開発

23G360　樋口史弥（最所研究室）

1　はじめに

ファイルのバージョン管理は，ユーザの誤操作によるデータの損失からデータを保護するために重要な機能である．ファイルサーバの中にはWAFLなどのスナップショットを作成できる従来のファイルシステムが提供するスナップショット(SS)作成機能を用い，一定間隔でSSを作成することでファイルのバージョン管理を行うものがある．これらのファイルシステムや論理ボリュームマネージャは，SSを作成する毎にすべてのファイルの差分データのみを保持する．これにより，ディスク使用量を抑制しつつファイルのバージョン管理を行える．

しかし，WAFLはファイルシステムボリュームレベルでSSを作成する．そのため，ユーザが必要としないタイミングでのファイルの差分データや，不必要なファイルの差分データが保持される．

このような不要な差分データは，ファイルサーバのような複数人が利用する環境において，ファイルサーバの各ユーザが自由にSSを作成した場合に多く生成される．そのためファイルサーバのユーザはSSを作成できない．このことから，ファイルサーバの各ユーザがSSを作成するためには，SS作成時に必要な差分データのみを保持することで，ディスク使用量の増加を抑制することが課題となる．

2　課題と要件

2.1　LFS(WAFL)のスナップショット作成の概要

WAFL(Write Anyware File Layout)はログ構造ファイルシステムであり，ファイルシステムボリュームのSSを作成できる．

図1はWAFLにおけるSS作成の動作を表している．図1 (a)はSSの作成前の状態を表している．Active filesystemはユーザが読み書きを行うことができるファイルシステムボリュームであり，ファイルA，B，Cの3つのファイルを保持している．図1 (b)はSS作成後の状態を表している．SSはActive filesystemの任意の時点の仮想的なコピーを作成することで作成される．作成直後はActive file systemとSSとでファイルA，B，Cの3つのファイルを共有している．図1 (c)はファイルB，Cの更新後の状態を表している．Active filesystemはファイルA，B’，C’を保持しており，ファイルB’，C’はファイルB，Cの差分データである．

この時，ユーザがファイルBの新たなバージョンを作成することを目的としてSSを作成した場合，ファイルC’の参照する差分データは不必要である．このような不要な差分データは，ファイルサーバのような複数人が利用する環境において，ファイルサーバの各ユーザが自由にSSを作成した場合に多く生成される．そのため，多くのファイルサーバでは，管理者が定期的にSSを作成する．しかし，このような管理方法の場合，目標復旧時点(Recovery Point Objective)を達成できないファイル存在する場合がある．そのため，ファイルサーバの各ユーザが自由にスナップショットを作成することで，ファイルのバージョンを作成できることが理想である．そのためには，スナップショット作成時に必要な差分データのみを保持することで，ディスク使用量の増加を抑制することが課題となる．

2.2　YuihaFS の要件

要件 1: 選択的バージョン作成　SSを作成できる従来のファイルシステムはファイルシステムボリュームレベルのSSを作成する．これにより不必要な差分データが保持される．不必要な差分データの保存を避けるために，YuihaFSは特別なインタフェースを用いず，任意のファイルのバージョンを任意のタイミングで作成できる機能を提供する．

要件 2: 試行錯誤への対応　ドキュメントやソースコードはユーザによって試行錯誤される．試行錯誤を行うためには，以前に作成したバージョンの内容を保持しつつ編集できる必要がある．しかし，従来のSSを作成できるファイルシステムはSSに対して書き込みを行えない．そのため，以前のバージョンを編集する場合，以前のバージョンが保持するデータを新たなファイルにコピーした後，編集を加える必要がある．その結果，ファイル数が増えることで，不必要な差分データ量が増加する．そのため，YuihaFSでは以前のバージョンの内容を保持しつつ，書き込みを行える機能を提供する．

要件 3: バージョン削除　時間の経過とともに，いくつかのバージョンは不必要になる．不必要な差分データを削減するために，従来のSSを作成できるファイルシステムでは，ファイルシステムボリュームレベルのSSの削除機能を提供している．一方，

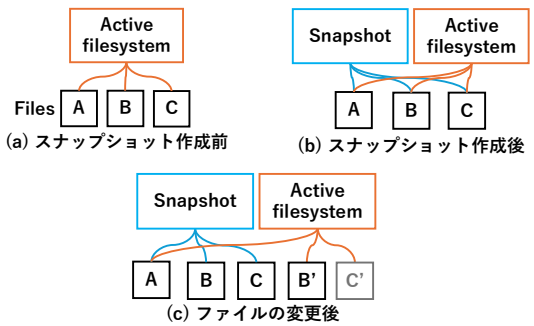

図 1: WAFLにおけるスナップショット作成の動作

YuihaFS はファイルごとにスナップショットのバージョンを作成する．そのため，YuihaFS では，ユーザやアプリケーションが任意のバージョンを削除するためのインタフェースを提供する．

3 YuihaFS の設計

3.1 選択的バージョン作成

要件 1 を満たすために，バージョン作成を制御するためのインタフェースを提供する．ファイルの重要度に応じてバージョン作成を制御するために，バージョン作成指示フラグ(*O_VERSION*)を *open* システムコールに導入する．このフラグが open システムコールの第 2 引数に設定された場合，新たなバージョンを作成する．

新たに作成されたバージョンは親バージョンのデータブロックの参照をコピーすることで祖先バージョンとデータブロックを共有する．共有されているデータブロックへの書き込みは Copy-On-Write(CoW)を行う．

3.2 試行錯誤への対応

要件 2 を満たすために，YuihaFS はバージョンの分岐を作成する．これにより，ユーザはバージョンを木構造として管理できる．

また，各バージョンに対して許可されている操作は，子バージョンの有無によって異なる．子バージョンを持たないバージョンを Leaf バージョンと呼び，読み書きが許可されている．一方，1 つ以上の子バージョンを持つバージョンを Trunk バージョンと呼び，誤った書き込みによってファイルの変更履歴が上書きされてしまうことを防ぐために，Trunk バージョンに対しては読み出しのみが許可されている．しかし，ユーザが試行錯誤を行うために，Trunk バージョンを修正する必要がある．そのため，書き込み可能なモードで Trunk バージョンを開いた場合，開いた Trunk バージョンを親バージョンに持つ新たな Leaf バージョンを作成することで対応する．

3.3 バージョン削除

要件 3 を満たすために YuihaFS はユーザやアプリケーションによって指定されたバージョンを削除する機能を提供する．バージョン削除は、主に 2 つの手順で構成される．まず，削除対象バージョンへの参照がバージョンの木構造から削除される．次に，削除されたバージョンによってのみ参照されるデータブロックが解放される．

4 評価

要件 1 である選択的バージョン作成の効果を評価するために，YuihaFS とログ構造ファイルシステムである nilfs のファイルシステムボリューム内の差分データ量を比較する．また，バージョン作成による書き込みスループットへの影響を評価するために，YuihaFS と Ext3 の書き込みスループットを比較する．YuihaFS に保存されるファイルとしてドキュメントを想定している．ドキュメントはエディタによって上書きされる．そのため，シーケンシャル上書きの実験を行う．シーケンシャル上書き実験では 500KB，1MB，2MB，4MB の書き込みを行う．また，YuihaFS のバージョン作成制御の効果を評価するために，バージョン作成頻度を変えた場合の実験も行う．そのため，1 つのファイルに 100 回の書き込みを行い，nilfs は書き込み毎に SS を作成する．一方，YuihaFS は nilfs と同じ割合(100%)，半分の比率(50%)，4 分の 1 の比率(25%)，バージョン作成無し(0%)の頻度でバージョンを作成する．

図 2 はシーケンシャル上書きの差分データ量を表している．YuihaFS のバージョン作成割合を抑えると，差分データ量も減少している．YuihaFS のバージョン作成割合を 50%，25%に設定することで，YuihaFS の差分データ量は nilfs と比べて 50%，75%削減できた．これは，ファイルへの上書きにより生じる CoW によって生成される差分データを削減できたためである．

図 3 はキャッシュを有効化した場合のシーケンシャル上書きの書き込みスループットを表している．4MB 上書き時の Ext3 に対する YuihaFS の書き込みスループットの割合は，YuihaFS の利用法として想定しているバージョン作成割合 25%の場合は 15%ほどの低下に抑えた．Git のコミット作成を YuihaFS のバージョン作成と対応付けると，実際のバージョン作成頻度は 25%よりも低いと考えられる．そのため，バージョン作成による書き込みスループットの低下は小さく抑えられる．

5 終わりに

ファイルごとにスナップショットを作成できるファイルシステム YuihaFS の開発と評価を行った．YuihaFS のバージョン作成割合を nilfs の半分に減らすことで，nilfs に対する YuihaFS の差分データ量を 50%削減することができた．

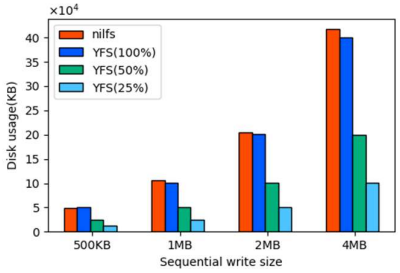

図 2: シーケンシャル書き込みの差分データ量

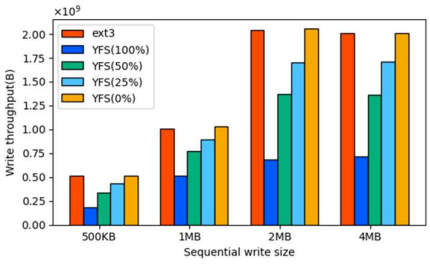

図 3: シーケンシャル書き込みの書き込みスループット