B-023

ライブCDへ向けたExt2ファイルシステムの書込みアルゴリズムの検討と評価

Examination and Evaluation of Writing Algorithm of Ext2fs for Live CD

北川 健司† 丹 英之† 千葉 大作† 須崎 有康‡ 飯島 賢吾‡ 八木 豊志樹‡
Kenji Kitagawa Hideyuki Tan Daisaku Chiba Kuniyasu Suzaki Kengo Iijima Toshiki Yagi

1. 概要

ライブ CD KNOPPIX 日本語版[1]では，User-Mode Linux の CopyOnWrite(以後 COW)機能を使用するため，ルートイメージのファイルシステムが iso9660fs から Ext2fs へと変更された[2]．しかし，Ext2fs はファイル断片化の影響を緩和する設計となっており，CD のようなライトワンスで且つ，ランダムアクセスを不得手とするデバイスの場合はシーク時間が顕著となり不向きである．本論文では，Ext2fs をライブ CD 向けのファイルシステムとして改変し，その有効性について報告する．

2. KNOPPIX[3]とは

KNOPPIX とは，ドイツの Klaus Knopper 氏により Debian GNU/Linux をベースに開発された 1CD 起動 Linux であり，産業技術総合研究所(AIST)で日本語化，及び配布が行われている．以下に挙げる KNOPPIX の特徴により，Linux 中上級者だけの利用に留まらず，Linux の使用を躊躇していたユーザに対しても安心と手軽さを提供し，Linux に触れるための敷居を低くしている．

- 面倒なインストール作業が不要
- 既存の HD ドライブ環境を破壊しない
- 致命的な問題が発生してもリブートで初期状態
- 安価に大量にプレス可能
- 目的に合わせたテンポラリ OS (復旧用，アプリ別)
- ハードウェアの自動認識・設定 (Autoconfig)
- 圧縮ループバックデバイス (cloop)

3. cloop(CompressedLoopbackBlockdevice)とは

ループバックデバイス(loop)のドライバは，ファイルを HD ドライブのようなブロックデバイスとして扱う機能を提供する．cloop とは，このブロックが圧縮されたものを扱えるように拡張されたループバックデバイスのことである．KNOPPIX では，ルートファイルシステムを cloop ファイルとして扱うことで，約 1.8GB の情報を CD メディア 700MB に格納することができる．cloop 上に格納するファイルシステムは何でも良く，KNOPPIX 本家では，iso9660fs が採用されている．

4. ルートファイルシステムの変更

KNOPPIX 日本語版では，User-Mode Linux に擬似的に書込みを行う機能である COW を使用するため，cloop 上のファイルシステムを ReadOnly な iso9660fs から，ブロック単位で Read/Write を行う Ext2fs に変更した．

†(株) アルファシステムズ，Alpha Systems Inc.
‡(独) 産業技術総合研究所，
National Institute of Advanced Industrial Science and Technology (AIST)

この変更により，COW によってユーザへルートファイルシステムへの書き込みを提供する OBD[4]が実現された．Ext2fs ではファイルシステムの一部の更新がファイルシステムイメージ全体に影響しない．このため，分割圧縮ブロックファイルを用いる HTTP-FUSE[5]でも，ユーザが変更を加えた内容を含むブロックファイルだけを差し替えることでファイルシステムイメージ全体が更新されることになる．

5. Ext2fs をライブ CD で使用することの問題点

Linux のファイルシステムとして一般的に用いられてきた Ext2fs は，フラグメンテーションが起こりにくいと言われてきた．

図 1 は KNOPPIX3.8.1 日本語版の cloop 上に格納された Ext2fs イメージを DAV(Disk Allocation Viewer)[6]で可視化したものである．一度目の書き込みであるにもかかわらず，ファイルの断片化が随所に見受けられた．

これは，書き込むデータ量にあわせてファイルシステムのイメージサイズが作成されており，空き領域が無くなる程，サイズの大きなファイルが断片化するからである．

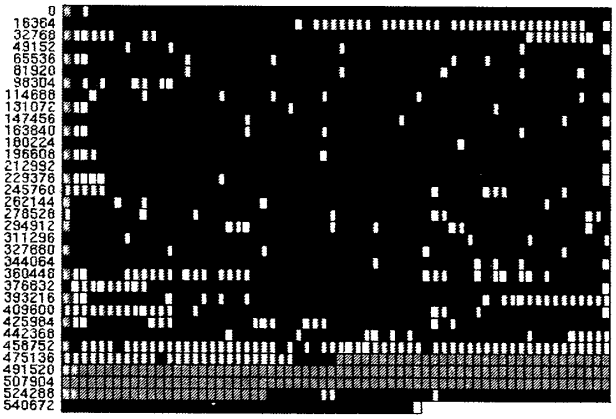

図 1．KNOPPIX イメージのフラグメントの様子

元来 Ext2fs は HD ドライブのような書換えが頻繁に起こるデバイスを想定しており，随所に分散と連続配置をある程度考慮した(フラグメンテーションが起きにくく，アクセス速度が速くなるようなブロック配置)設計となっている．それに対し，ライトワンスメディアに書込みを行う場合は，書き換え処理が絶対に発生しないため，データブロックを各ブロックグループに分散させたり，プリアロケートしておくなど，将来のフラグメンテーションを見越したアルゴリズムを利用する必要性がない．

今回の KNOPIX イメージの表示によって，ファイルの非連続配置やスパースなブロック配置による読み込み効率の低下，関連するディレクトリツリーの分散などが発

生し，HD ドライブでは無視できたシーク時間が，ランダムアクセスの不得手な CD ドライブでは顕著に現れると考える．また，ファイルシステムに iso9660fs を使用する場合，mkisofs コマンドに-sort オプションを付加することで，指定順にファイルを並べることが出来るが，Ext2fs では，不可能となっている．

6. ファイルシステム Ext2kai の作成

Ext2kai とは，既存の Ext2fs をベースに，前項で挙げたライトワンスの場合には必要で無いと思われるアルゴリズムについて改変を加えたファイルシステムである．アルゴリズムの改変内容は，以下の通りである．

- ディレクトリ作成時，各ブロックグループの平均 i ノード使用率に因らず先頭のブロックから作成する
- ファイル作成時，親ディレクトリの存在するブロックグループ内のランダムブロックに作成せず，先頭ブロックから詰めて作成する
- ファイル作成時，将来のファイル拡張に因るフラグメンテーションを防ぐための余裕を持ったブロック確保を行うプリアロケート機能を排除
- ディレクトリのコピー時，ディレクトリサイズに応じて，コピー先でデータブロックをプリアロケートする機能を追加

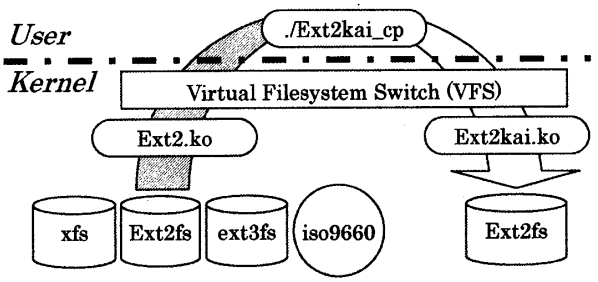

図 2．Ext2kai を経由したデータの流れ

Ext2kai を用いることで，各ファイルシステムからのコピー時にすべてのデータブロックをシーケンシャルに隙間なく詰めることで，フラグメンテーションを防ぐ事が出来る．

ただし，ディレクトリのコピーは，コピー先で mkdir されるのみであるため，そのディレクトリへファイルがコピーされる度にファイル名や i ノード番号などの更新が行われ，ある一定数以上のファイルがコピーされるとデータブロックが足りなくなり，新規にデータブロックを確保する．この時ディレクトリのフラグメンテーションが発生する．この問題に対処するため，新たなファイルコピーコマンド Ext2kai_cp を用意し，ディレクトリコピーがされる際にそのディレクトリサイズを Ext2kai に渡すことで，データブロックをプリアロケートする仕組みを考案した．図 2 にそのデータの流れを示す．

本研究では，Ext2kai ファイルストア(make menuconfig により選択可)と Ext2kai_cp コマンドを作成した．それぞれ kernel2.6.11 の Ext2fs と coreutils-5.2.1 の cp をベースに開発を行った．

7. 評価

前項で開発した Ext2kai のカーネルモジュールを KNOPPIX3.8.1(kernel2.6.11)に実装し，フラグメンテーションの検証と特定ファイルを読み出す実験を行った．検証した PC は，CPU:Pentium4 2GB，RAM:512MB，CD-Drive:40 倍速である．

まず，DAV によるフラグメンテーション解析を行った．その結果，フラグメンテーションはシステムブロックによる分断のみであり，DAV 開発者によるフラグメンテーションの定義では，フラグメンテーションは発生していないことになる．

次に，番号を振ったバイナリファイル(1KB) 5000 個を，剰余の定理に基づき 50 個のディレクトリにそれぞれ 100 個ずつ配置し，番号順にすべてのファイルを読み出す実験を行った．実験用 KNOPPIX は，通常通りの手順で作成したもの(Ext2fs)と，ファイルを指定順に並べて作成したもの(Ext2kai)の 2 枚を準備した．その結果，最大 172(/sec)，平均 34.6(/sec)の速度差が計測出来た．読み込み総量が 5MB で平均約 13%の読み込み時間が削減された．

8. まとめと今後の展開

本論文では，Ext2fs をベースにライブ CD に特化したファイルシステム Ext2kai とユーティリティツールの作成を行った．フラグメンテーションは全く発生しておらず，特定ファイルの読み出し速度は，全容量 5MB であるブロックグループに比較的集約していたにも関わらず，既存の読み出し速度より，最大 172(/sec)，平均で 34.6(/sec)のアクセスの向上が計測できた．今回は，CD からの特定ファイルの読み出しに焦点を絞ったが，今後は，起動時に使用するファイルや巨大なアプリケーションで使用するファイルなどトランザクション単位にファイルを並べることで，シーク時間の低減，cloop ブロック辺りの有効バイト率の向上，圧縮ブロック伸張時間やメモリの有効利用に繋がると考えている．

参考文献 & URL

[1] KNOPPIX 日本語版
http://unit.aist.go.jp/itri/knoppix/index.html
[2] 須崎, 飯島, 八木, 丹, "CopyOnWrite を適用した UML-KNOPPIX", FIT2004, B-027, 2004
http://unit.aist.go.jp/it/knoppix/FIT04-suzaki.PDF
[3] knoppix, "http://www.knopper.net/knoppix/"
[4] 丹, 須崎, 飯島, 八木, "CopyOnWrite を用いたオーバーレイブロックデバイスの実装", LinuxConference2005
http://lc.linux.or.jp/paper/lc2005/CP-05.pdf
[5] 須崎, 八木, 飯島, 丹, "HTTP-FUSE KNOPPIX", Linux Conference2005,
http://lc.linux.or.jp/paper/lc2005/CP-02.pdf
[6] 平松, 杉田, 藤原, " フラグメンテーション解析を支援する DAV と LKST Log Tools の開発", LinuxConference2005,
http://lc.linux.or.jp/paper/lc2005/CP-06.pdf