B-021

擬似不揮発性メモリを用いた OFF2F プログラムの評価
Evaluation of OFF2F Program Using Pseudo Non-Volatile Memory

額田 哲彰 † 佐藤 将也 †† 山内 利宏 ††† 谷口 秀夫 †††
Tetsuaki Nukata Masaya Sato Toshihiro Yamauchi Hideo Taniguchi

1. はじめに

揮発性メモリと不揮発性メモリ（Non-Volatile Memory：以降，NV メモリ）を混載した計算機において，高速なプログラム実行を可能にする新たな実行ファイル形式（OFF2F：Object File Format consisting of 2 Files）を提案した [1]．また，FreeBSD 11.0-RELEASE（以降，FreeBSD）の動作環境で実現した擬似 NV メモリを利用して，OFF2F の有効性を明らかにした [2]．

本稿では，UnixBench 4.1.0（以降，UnixBench）を使用し，擬似 NV メモリを利用し実行した OFF2F プログラムの実行時間と FreeBSD 上で実行した ELF（Executable and Linkable Format）プログラムの実行時間の比較を述べる．

2. OFF2F プログラムの実行環境

2.1 OFF2F [1]

実行ファイルは，4 つの内容（ヘッダ部，テキスト部，データ部，関係情報部）から構成される．OFF2F は，実行ファイルが 2 つのファイルに分割されている形式であり，読み込みのみ発生するテキスト部のファイルを NV メモリ上に置く．このようにすることで，OFF2F プログラム実行時には，NV メモリ上のファイルをそのまま仮想メモリ空間にマッピングできるため，テキスト部の外部記憶装置からの読み込み時間を短縮し，プログラム実行を高速化できる．

2.2 擬似 NV メモリ [2]

実メモリの一部を OS が使用しない領域とし，擬似 NV メモリとして扱う．この領域は，OS が使用しない領域であるため，OS の初期化処理でも利用されず，計算機の電源を OFF しない限り OS を再立ち上げしても，内容は保存され更新されない．つまり，NV メモリとして擬似できる．

擬似 NV メモリに格納した OFF2F プログラムの様子を**図 1** に示す．OFF2F プログラムのヘッダ部を含むファイルは外部記憶装置上に存在し，テキスト部のファイルは擬似 NV メモリ上に存在する．OFF2F プログラム実行時は，テキスト部として擬似 NV メモリ上のファイルを利用する．

2.3 プログラム実行時の処理流れ

OFF2F プログラム実行時の処理流れは，ELF プログラムの場合と大きく 2 つの処理が異なる．1 つは，テキスト部のページ例外（PF：Page Fault）処理である．ELF プログラム実行時にページ例外が発生したとき，外部記憶装置から当該データを実メモリに読み込み，実メモリをマッピング表に登録する．一方，OFF2F プログラム実行時に擬似 NV メモリ上データに対するページ例外が発生したとき，当該の擬似 NV メモリのページをマッピング表に登録する．

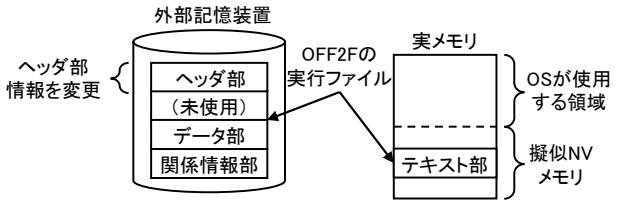

図 1 擬似 NV メモリに格納した OFF2F プログラム

もう 1 つは，exec によるプログラム起動時にファイルの先頭を読み込み，ヘッダ部情報を獲得する処理である．ELF プログラムでは，ヘッダ部とテキスト部は連続して存在し，どちらも読み込みのみ行われる部分であるため，先頭 64 KB を読み込むことでページ例外の発生回数を削減している．一方，OFF2F プログラムでは，ヘッダ部はデータ部と連続して存在し，データ部は読み書きされる部分であるため，先頭 4 KB を読み込む．

3. 評価

3.1 評価の環境

評価には Intel Core i3-8100（3.1 GHz）と 8 GB のメモリを搭載した計算機を使用した．なお，OS の実メモリを 7 GB とし，擬似 NV メモリを 1 GB とした．OS は FreeBSD を使用し，外部記憶装置には 500 GB の HDD（5,400 RPM）を使用した．また，ベンチマークの実行前に 2 GB のデバイスファイルを読み込む処理を行い，HDD の内部に搭載されているキャッシュの影響を抑制した．ベンチマークは UnixBench を使用し，テキスト部でページ例外が多数発生する場合 [2] として C コンパイラの実行回数を測定し，exec を多数実行する場合として execl() の呼び出し回数を測定した．

3.2 C コンパイラの実行回数

UnixBench の 測定プログラムは 1 分間，ループ内で execvp() を呼び出し，/usr/bin/cc（以降，cc）を実行する．cc はテキスト部の大きさが約 42,796 KB のプログラムであり，本測定では C 言語で書かれた約 3.5 KB のソースファイルをコンパイルする．UnixBench は測定プログラムを 3 周実行し，各周における 1 分あたりの cc の実行回数，およびベンチマークの結果として幾何平均を算出する．また，各周における 1 回目の cc の実行時間も測定するように UnixBench を改造した．cc が ELF プログラムの場合と OFF2F プログラムの場合で測定を

† 岡山大学大学院自然科学研究科, Okayama University
†† 岡山県立大学情報工学部, Okayama Prefectural University
††† 岡山大学学術研究院自然科学学域, Okayama University

表 1　UnixBench の C コンパイラの実行回数

	1 周目 (回/min)	2 周目 (回/min)	3 周目 (回/min)	結果
ELF	2,854.6	2,981.0	2,982.0	2,938.6
OFF2F	2,952.0	2,976.0	2,974.0	2,967.3

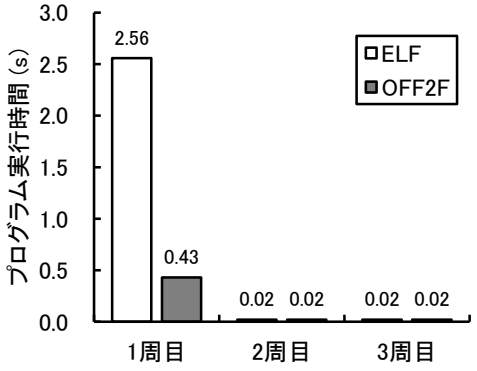

図 2　各周における 1 回目の cc の実行時間

表 2　UnixBench の execl() の呼び出し回数

	1 周目 (回/s)	2 周目 (回/s)	3 周目 (回/s)	結果
ELF	3,784.9	3,785.7	3,790.3	3,787.0
OFF2F	3,735.8	3,743.4	3,740.3	3,739.8

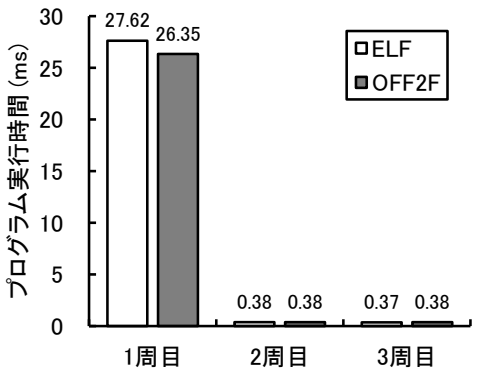

図 3 各周における 1 回目の測定プログラムの実行時間

行った．

ベンチマークの結果を**表 1** に示す．また，各周における 1 回目の cc の実行時間を**図 2** に示す．表 1 と図 2 から，以下のことが分かる．

(1) 表 1 より，ベンチマークの結果は，OFF2F プログラムの方が約 1.0 %（$(2,967.3-2,938.6)/2,938.6$）良い．各周の実行回数を見ると，2 周目と 3 周目では実行回数の差が僅かである一方，1 周目では OFF2F プログラムの場合で実行回数が約 3.4 %（$(2,952.0-2,854.6)/2,854.6$）多い．

(2) 図 2 より，1 周目の 1 回目において，OFF2F プログラムの実行時間は，ELF プログラムの実行時間の約 6 分の 1（0.43/2.56）に短縮できる．これは，ページ例外が発生した際に，OFF2F プログラムによってテキスト部の外部記憶装置からの読み込み時間を短縮したためである．この結果，OFF2F プログラムの実行時間を短縮できるため，項目 (1) に示す 1 周目の効果が生まれる．

(3) 図 2 より，ELF プログラムと OFF2F プログラムで，2 周目の 1 回目と 3 周目の 1 回目の cc の実行時間は同じであり，この実行時間は OFF2F プログラムにおける 1 周目の 1 回目の実行時間よりも短い．これは，cc の内容がメモリ上にキャッシュされ，1 周目の 2 回目以降の実行では，外部記憶装置からの読み込みを省略するためである．

3.3 execl() の呼び出し回数

UnixBench の測定プログラムは execl() を呼び出し，自身と同じ測定プログラムを実行することで，execl() を 30 秒間繰り返す．UnixBench は測定プログラムを 3 周実行し，各周における 1 秒あたりの execl() の呼び出し回数，およびベンチマークの結果として幾何平均を算出する．また，各周における 1 回目の測定プログラムの実行時間も測定するように UnixBench を改造した．測定プログラムはテキスト部が約 10.7 KB のプログラムであり，このプログラムが ELF プログラムの場合と OFF2F プログラムの場合で測定を行った．

ベンチマークの結果を**表 2** に示す．また，各周における 1 回目の測定プログラムの実行時間を**図 3** に示す．表 2 と図 3 から，以下のことが分かる．

(1) 表 2 より，ベンチマークの結果は，ELF プログラムの方が約 1.2 %（$(3,787.0-3,739.8)/3,787.0$）良い．これは，テキスト部について，ELF プログラムではプログラム起動時に読み込むためページ例外が発生しない一方で，OFF2F プログラムではページ例外が発生するためである．

(2) 図 3 より，1 周目の 1 回目において，OFF2F プログラムの実行時間は，ELF プログラムの実行時間より約 4.6 %（$(27.62-26.35)/27.62$）短い．これは，プログラム起動時に，OFF2F プログラムによってテキスト部の外部記憶装置からの読み込み時間を短縮したためである．

4. おわりに

UnixBench を使用し，擬似 NV メモリを利用し実行した OFF2F プログラムの実行時間と FreeBSD 上で実行した ELF プログラムの実行時間の比較を述べた．評価により，UnixBench の結果は，C コンパイラの実行回数の場合は OFF2F プログラムの方が約 1.0 %良く，execl() の呼び出し回数の場合は ELF プログラムの方が約 1.2 %良い結果となった．一方で，1 回目の実行時間に着目すると，テキスト部の外部記憶装置からの読み込み時間の短縮により，C コンパイラが OFF2F の場合に実行時間を約 6 分の 1 に短縮できることを示した．

謝辞　本研究の一部は，JSPS KAKENHI 21K11830 による．

参考文献

[1] Sato, M. and Taniguchi, H.: OFF2F: A New Object File Format for Virtual Memory Systems to Support Volatile/non-Volatile Memory-Mixed Environment, *International Journal of Machine Learning and Computing*, Vol. 9, No. 4, pp. 387–392 (2019).

[2] 高杉　頌，額田哲彰，佐藤将也，谷口秀夫：揮発性/不揮発性メモリ混載計算機において実行プログラムを分散配置する OFF2F プログラムの性能評価，情報処理学会研究報告，Vol. 2020-DPS-185, No. 9, pp. 1–7 (2020).