社団法人 情報処理学会 研究報告
IPSJ SIG Technical Report

2004－ARC－160 (18)
2004／12／2

キャッシュ・ミス頻発命令とその特徴解析

堂後 靖博† 三輪 英樹‡ ヴィクトル・マウロ・グラール・フェヘイラ‡
井上 弘士‡ 村上 和彰‡
†福岡大学大学院工学研究科電子情報工学専攻 ‡九州大学大学院システム情報科学研究院
arch-ccc-lpc@c.csce.kyushu-u.ac.jp

あらまし メモリ・ウォール問題(プロセッサー主記憶間の性能差拡大)を解決する有効な手段の1つとして，Delinquent命令の活用がある．例えば，Delinquent命令のアドレス計算用コードを別スレッドとして生成し，これを投機実行する事でプリフェッチ精度を向上できる．しかしながら，プロセッサー主記憶間の性能差は依然として拡大の一途を辿っており，Delinquent命令に着目したより効果的な高性能化方式の確立が望まれる．そこで本稿では，D命令に関する特徴解析を行う．具体的には，Delinquent命令の発生頻度や入力依存性，生存区間，アクセス・パタン等について調査する．本研究で得られた結果は，Delinquent命令に基づくメモリ性能高性能化技術開発の基礎データとして用いる事ができる．

Characteristic Analysis of Delinquent Memory Access Instructions

Yasuhiro DOUGO† Hideki MIWA‡ Victor Mauro Goulart FERREIRA‡
Koji INOUE‡ and Kazuaki MURAKAMI‡
† Dept. of Elec. and Computer Science, Fukuoka University ‡ Dept. of Informatics,Kyushu University
arch-ccc-lpc@c.csce.kyushu-u.ac.jp

Abstract Recent remarkable advances of VLSI technology have been increasing processor speed and DRAM capacity dramatically. However, the advances also have introduced a large and growing performance gap between the processor and DRAM, this problem is referred to as " Memory Wall" , resulting in poor total system performance in spite of higher processor performance. In order to solve this problem, researchers have been proposed high-performance techniques to alleviate the effect of delinquent memory-access instructions. In this paper, we investigate the detail of behavior of the delinquent memory-access instructions. The results presented in this paper will be useful to develop new approaches against the memory wall problem.

1. はじめに

メモリ・ウォール問題（プロセッサー主記憶間の性能差拡大）[2]を解決する有効な手段の1つとして，Delinquent命令（以下，D命令と記す）の活用がある．ここでD命令とは，キャッシュ・ミスに伴う主記憶アクセスを頻繁に引き起す静的メモリ・アクセス命令の事である．多くのキャッシュ・ミスは極めて少数の（静的な）メモリ・アクセス命令によって引起される事が知られている．例えば，幾つかのベンチマークでは，L1キャッシュ・ミスを引き起す上位10個の静的ロード命令（つまりDロード命令）が全ミス回数の90%以上を占める[1]．これまでに，D命令に特化した最適化を行うことで効率的にメモリ性能を向上させる様々な技術が提案された．例えば，D命令のアドレス計算用コードを別スレッドとして生成し，これを投機実行する事でプリフェッチ精度を向上する方式などがある[1,3]．

一般に，プログラム実行の振舞いはプロセッサ構成やメモリ構成，さらには，入力データに依存する．そのため，あるプログラムが与えられた際，必ずしも固定的にD命令を決定できる訳ではない．また，プログラム実行においても，各D命令の振舞いは大きく異なると予測される．プロセッサー主記憶間の性能差は依然として拡

大の一途を辿っており，D 命令に着目したより効果的な高性能化方式の確立が望まれる．そのためには，D 命令の特徴を詳細に分析・解析し，実行振る舞いに適した最適化方式を選択する必要がある．

本稿では，高性能プロセッサを想定し，D 命令に関する特徴解析を行う．具体的には，D 命令の発生頻度や入力依存性，生存区間，アクセス・パタン等について調査する．本研究で得られた結果は，D 命令に基づくメモリ高性能化技術開発の基礎データとして用いる事ができる．なお，本稿では L2 キャッシュまでがオンチップ化されている場合を想定する．また，以降，「キャッシュ・ミス」とは，オフチップ・アクセスを引起す L2 ミスを意味する．

本稿は以下のような構成である．第 2 節では D 命令に関する解析項目を示す．第 3 節では実験環境について述べ，第 4 節で特徴解析結果を示す．第 5 節では得られた実験結果に対して考察を行い，最後に第 6 節で本稿をまとめる．

2. 解析項目

本稿では，以下に示す 5 つの項目について D 命令の特徴解析を行う．

1. **発生頻度**
   D 命令が全キャッシュ・ミスに占める割合を調査する．本稿では，あるプログラムを実行した際，最も多くキャッシュ・ミスを発生する上位 10 個の静的メモリ参照命令を D 命令とする．
2. **同一性**
   ある入力データで検出された D 命令が，他の異なる入力データの場合でも同様に検出されるか調査する．
3. **ミス発生頻度の時間的変動**
   各 D 命令が引き起こすミス回数の時間的な変動を解析する．具体的には，各 D 命令において，各時間間隔におけるミス発生回数状況等を調べる．
4. **生存区間**
   プログラム実行において，D 命令がミスを発生させる期間を調べる．
5. **アクセス・パタン**
   D 命令がミスを発生させた際のアクセスするデータ・アドレスを取得し，アクセス・パタンを調査する．

表 1：シミュレータ上のプロセッサ構成

命令フェッチ		
命令フェッチ/デコード/発行幅	8	
RUUエントリ数	64	
LSQエントリ数	32	
分岐予測		
分岐予測	PHT: 8Kエントリ/履歴長11のgshare	
分岐先アドレス	BTB: 2Kエントリ4way、RAS: 32エントリ	
演算ユニット		
	int	fp
ALU数	4	4
実行レイテンシ	1	2
発行レイテンシ	1	1
MUL-DIV数	1	1
キャッシュ		
L1命令	32KB(64B/lines,1way,512lines)	
L1データ	32KB(64B/lines,2way,256lines)	
L2命令データ統合	2MB(64B/lines,4way,8192lines)	
オフチップ・アクセス・メモリバス幅	8B	
ヒット・レイテンシ		
L1命令	1クロック	
L1データ	2クロック	
L2命令データ統合	16クロック	
オフチップ主記憶	250クロック	
TLB		
命令	1Mエントリ(4096B/エントリ4way,256エントリ/way)	
データ	1Mエントリ(4096B/エントリ4way,256エントリ/way)	
命令/データ・ミスペナルティ	30	

以降，プログラム実行において，n 番目に多くのキャッシュ・ミスを引き起こした D 命令を Dn と表記する．例えば，D1 は最も多くのミスを発生させた D 命令である．

3. 実験環境

キャッシュ・ミス頻発命令の特徴解析を行うため，SPEC CPU 2000 ベンチマーク・セット [3] を用いた評価を行った．入力データとしては，

- small input（以降，入力 S と記す）
- reference input（以降，入力 R と記す）
- training input（以降，入力 T と記す）

の 3 種類を使用する．また，プロセッサ・シミュレータとしては，SimpleScalar ツール・セット [4] を用いる．本実験で想定したプロセッサ構成を表 1 に示す．なお，シミュレーション時間を短縮するため，入力 R ならびに入力 T に関しては，プログラム実行開始時から 20 億命令をフォワード，その後の 2 億命令のみを解析対象とした．一方，入力 S に関しては全実行を完了する．命令発行方式は out-of-order としている．

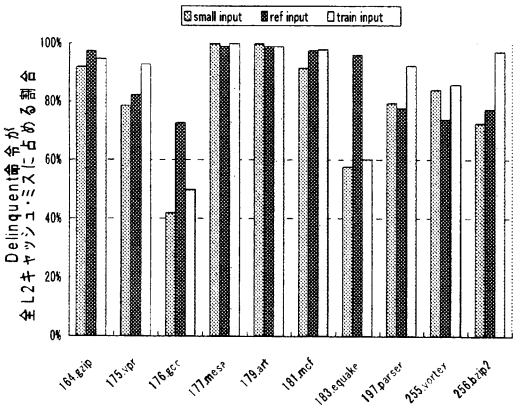

図 1：Delinquent 命令発生頻度

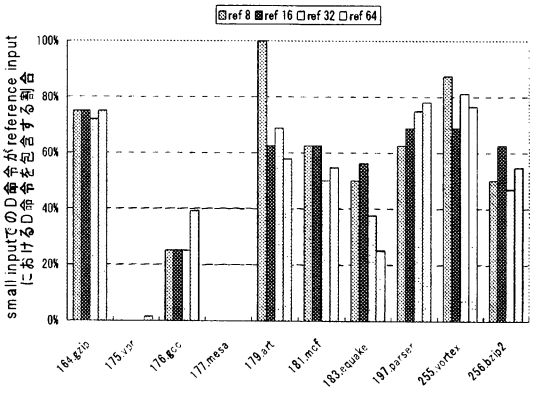

図 2：small input での D 命令が reference input における D 命令を包含する割合

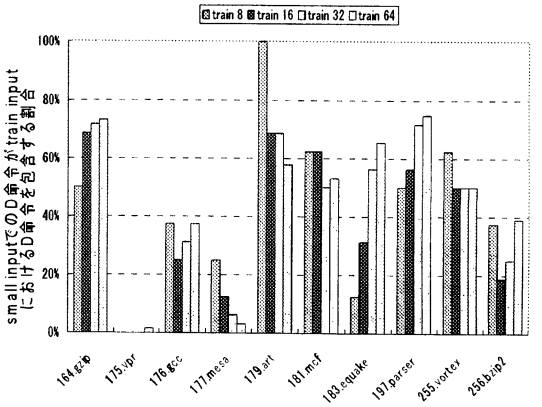

図 3：small input での D 命令が training input における D 命令を包含する割合

4. Delinquent 命令の特徴解析

本節では，D 命令の特徴解析結果について報告する．

4.1. 発生頻度

D 命令に特化したメモリ性能向上技術の有効性は D 命令の出現頻度に大きく依存する．そこで，各ベンチマークにおいて，異なる入力データを用いた際の D 命令出現頻度を調査した．全キャッシュ・ミスに D 命令が占める割合を図 1 に示す．

まず，プログラムの違いが D 命令出現頻度に与える影響を考察する．図 1 より，176.gcc と 183.equake を除く全てのベンチマークにおいて，わずか 10 個の D 命令が全ミスの 60%以上を占めている事が分かる．特に，177.mesa などの幾つかのプログラムでは，D 命令の占める割合が 90%以上と極めて高い．この結果より，多くのプログラムにおいて D 命令は存在する事が分かる．

次に，各プログラムにおいて，入力データの違いが D 命令出現頻度に与える影響を考察する．図 1 より，幾つかのプログラム（164.gzip, 181mcf, 177.mesa, 179.art）では，入力データの種類に関係無く，全キャッシュ・ミスの約 90%以上が 10 個の D 命令で占められている事が分かる．一方，176.gcc において，入力 R では D 命令が約 70%以上を占めているのに対し，入力 S や入力 R では 50%未満の占有率である．つまり，入力 R に比べ，他入力データではミスを発生させる命令の局所性が低い．また 183.equake でも同様の結果である．しかしながら，多くのベンチマークでは，どの入力データに関しても全キャッシュ・ミスの 80%程度の占有率である．これらの結果より，多くの場合，入力データの違いが D 命令の存在に与える影響は小さいと考える．

4.2. 同一性

D 命令に特化した静的最適化を行う場合，対象となる D 命令を固定的に決定する必要がある．そこで各ベンチマークにおいて，入力の違いが検出される D 命令（つまり，最も多くミスを発生する静的メモリ参照命令）の種類に与える影響を調査した．具体的には，入力 S において検出された上位 8，16，32，64 個の D 命令を基準とし，他入力データを用いた場合に検出される D 命令がどの程度含まれているか（包含率）を

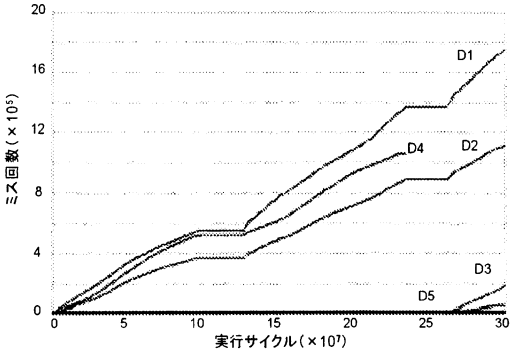

図 4：D1～D5 のミス発生状況(164.gzip)

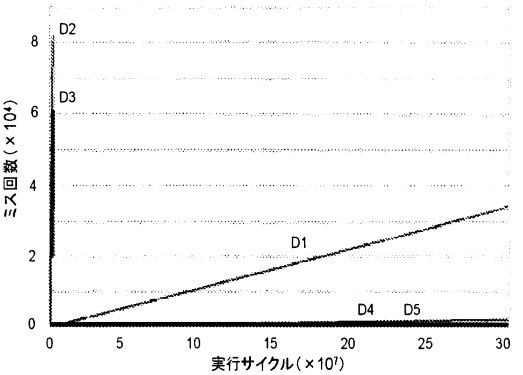

図 5：D1～D5 のミス発生状況(177.mesa)

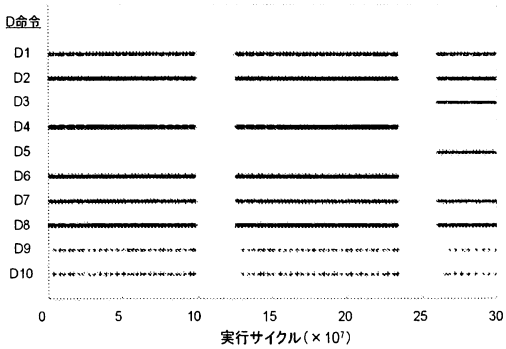

図 6：164.gzip の D 命令の生存区間(初期段階)

調べた．例えば，入力 S での D 命令 8 個が，入力 R での D 命令 8 個と全く同一の場合は包含率が 100%となる．

入力 R に関する包含率を図 2 に，入力 T に関する包含率を図 3 に示す． 175.vpr や 177.mesa では，入力 S で検出された D 命令の殆どが，入力 R や入力 T を用いた場合では検出されていない．これは，入力 S によって検出された D 命令と，入力 R や入力 T によって検出されたそれとは全く異なる事を意味する．一方， 164.gzip，179.art，197.parser，255.vortex のベンチマークでは，入力を変えた場合でも高い包含率となっている．しかしながら，多くの場合で包含率は 50%前後であり，入力データが，検出される D 命令の種類に与える影響は比較的大きいと考える．

4.3. ミス発生頻度の時間的変動

第 4.1 節ならびに第 4.2 節では，全プログラム実行終了後の統計データに基づき D 命令の特徴解析を行った．本節では，プログラム実行中における D 命令のミス発生頻度の変化を解析する．具体的には，入力 S での D1～D5 に関して，時間の経過と共にミス発生頻度が変化する様子を観測した．

164.gzipならびに 177.mesaに関して，D命令のミス回数累積を図 4 ならびに図 5 に示す．ここで，図中のDxとは，全プログラムの実行終了時に上位x番目に最も多くのミスを発生したD命令を表す．図 4 において実行サイクル 24（×10^7）で 2 番目にミス回数の多いD命令がプログラム終了時では 4 位(D4)となっている．またD3とD5 はこの時点での実行は発生していない．また，図 5 では，プログラム実行の極めて早い段階でD2 とD3 がミスを頻発させている．この時間では明らかにD2 が最も多くのミスを発生させている．またD2 とD3 はこの時点で実行は終了する．一方，プログラム実行の初期段階ではミス数がD2，D3 と比べ少ないD1 は，緩やかな傾きでミス回数を増加させ，プログラム終了まで実行される．

これらの結果より，各 D 命令のミス発生状況はプログラム実行と共に大きく変化する事が分かる．

4.4. 生存区間

第 4.1 節の実験結果より，多くのプログラムでは D 命令が存在する事が分かった．しかしながら，D 命令の生存期間が短い場合には，D 命令専用最適化の適用可能時間が短くなる．そこで，各 D 命令に関する生存期間を調査した．

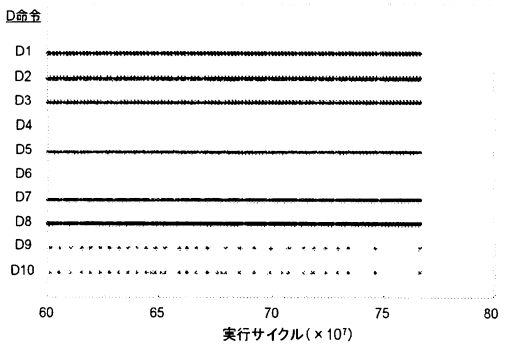

図 7：164.gzip の D 命令生存区間(終了段階)

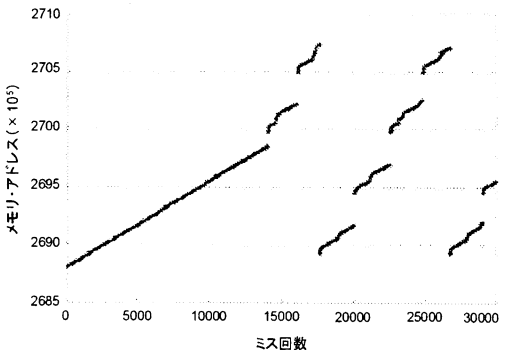

図 8：D 命令のアクセス・パタン(164.gzip D1)

164.gzip での D 命令発生状況に関して，プログラム実行開始時のプロットを図 6 に，プログラム実行終了時のそれを図 7 に示す．ここで縦軸は各 D 命令とし，横軸は実行サイクルである．本解析では，1 万サイクル中に 1 回でもミスが発生すると D 命令は生存していると判断する．図 6 および図 7 より，各 D 命令は比較的長期間に渡って実行されている（ミスを引き起こしている）事が分かる．しかしながら D4，D6 はプログラム実行の終了段階では実行は無く，早い段階で実行は終了している．このような命令に関しては，最適化の恩恵を受けることが期待できない．

4.5. アクセス・パタン

D 命令の特徴に適した最適化方式の選択はメモリ・アクセス・パタンに大きく依存する．そこで，各 D 命令のアクセス・パタンを観測した．164.gzip，176.gcc，ならびに，181.mcf における D1 のアクセス・パタンを図 8，図 9，ならびに，

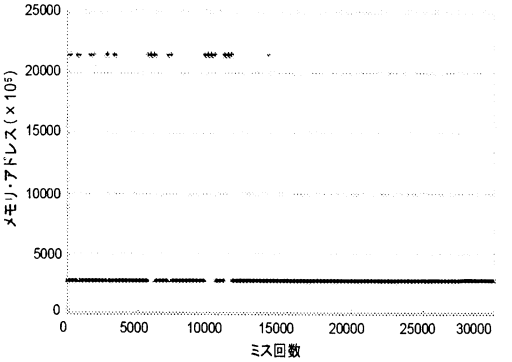

図 9：D 命令のアクセス・パタン(176.gcc)

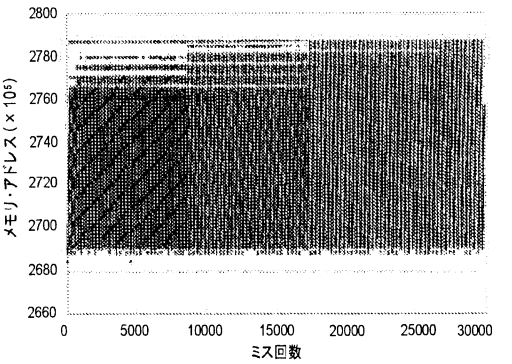

図 10：D 命令のアクセス・パタン(181.mcf)

図 10 に示す．ここで，縦軸は各 D 命令がアクセスしたアドレスを，横軸は発生したミス・カウント（通し番号）を表している．164.gzip の場合はアクセスするアドレス範囲は比較的広いが，周期的に同一アドレス範囲をアクセスする．176.gcc ではアクセス範囲が極めて局所的に集中している事が分かる．一方，181.mcf では，アクセス範囲が広く分散している．これらの結果より，D 命令の種類によって，アクセス・パタンは大きく異なることが分かる．

5. 考察

D 命令に基づきメモリ性能を向上するためには，1)D 命令を検出し，かつ，2)各 D 命令に対する最適化を実施しなければならない．本節では，第 4 節で得た実験結果に基づき，これら 2 つの課題について考察する．以下，第 4 節での解析結果をまとめる．

1. 実行対象となるプログラムや入力データが

異なる場合でも，その多くでD命令が存在する（第4.1節の結果より）．

2. 同一プログラムにおいて，入力データが異なると，D命令の種類（つまり，ミスを頻発する静的メモリ・アクセス命令）は異なる場合がある（第4.2節の結果より）．
3. あるプログラムの実行において，ミス発生頻度の時間的変動は各D命令によって異なる．（第4.3節の結果より）
4. 殆どの場合でD命令の生存区間は比較的長い．（第4.4節の結果より）
5. D命令のアクセス・パタンは様々であり，主に局所集中型，固定ストライド型，広域分散型に分類できる（第4.5節の結果より）．

D命令を検出するためには，プログラム実行に関する何らかのプロファイル情報を利用しなければならない．プロファイル情報の取得手段としては，静的プロファイリング（実行前にシミュレーション等により取得）と動的プロファイリング（実行時に取得）が考えられる．しかしながら，解析結果2で説明したように，検出されるD命令は入力データによって異なる場合がある．また，解析結果3で示すように，実行中にD命令のミス発生頻度は大きく変動する．したがって，このような実行時の振舞いに適応するには動的プロファイリングが有効であると考える．

次に，D命令に基づく最適化について考察する．解析結果1ならびに4より，多くのプログラムにおいてD命令が存在し，かつ，その生存区間は比較的長いことが分かる．これは，D命令に着目したメモリ性能向上技術が，多くの場合で性能向上を達成可能である事を意味する．また，解析結果3や5より，D命令のミス発生頻度ならびにアクセス・パタンは様々である事が分かった．したがって，多くのD命令に対処するには，固定的な最適化手法ではなく，各D命令の特徴に応じた手段を用いる方が有効であると考える．

6. おわりに

本稿では，D命令を考慮したメモリ高性能化技術開発の基礎データとして用いることを目的とした，D命令の特徴解析を行った．その結果，プログラムや入力データが異なる場合でも，その多くでD命令は存在し，同一プログラムにおいて，入力データが異なると，D命令の種類は異なる場合があることが分かった．また，実行中にD命令のミス発生頻度が大きく異なるためD命令を検出する際，実行振舞いに適した検出手法を施す必要があると分かった．また，各D命令の特徴が異なることより，アクセス・パタンは様々であった．そのため各D命令に適した最適化手法の適用が望まれる．今後，本研究で得られた特徴解析をもとに，D命令を考慮したメモリ高性能化技術の確立ならびに手法適用に対する性能評価を行う．

謝辞

多くの有用なアドバイスを頂いた福岡大学モシニャガ・ワシリー教授に感謝致します．また，本研究を進めるにあたり，多くのご指導を頂いた富士通株式会社の池田正幸氏，丸山拓巳氏，富士通研究所の勝野昭氏，坂本真理子氏に感謝致します．なお，本研究は一部，文部省科学研究費補助金（課題番号：14GS0218, 14702064,14102027）による

参考文献

[1] J. D. Collins, H. Wand, D. M. Tullsen, C. Hughes, Y. F. Lee, D. Lavery, J. P. Shen, "Speculative Precomputation: Long-range Prefetching of Delinquent Loads," Proc. of the 28th Int. Symposium on Computer Architecture, pp.14-25, June 2001.

[2] J.L.Hennessy and D.A.Patterson,"Computer Architecture:A Quantitative Approach 3rd Edition",Morgan Kaufmann Publishers,San Francisco,2003.

[3] A. Roth and G. Sohi, "Speculative Data-Driven Multithreading," Proc. of the 7th Int. Symposium on High-Performance Computer Architecture, pp.37-49, Jan. 2001.

[4] SimpleScalarLLC, http://www.simplescalar.com

[5] SPEC –Standard Performance Evaluation Corporation.