

PA-RISC用ステップ解析システムの開発

4G-7

高崎 繁夫、中原 雅彦、円光 豊、山田 公稔†、吉澤 康文

日立製作所システム開発研究所、†同 ソフトウェア開発本部

1. 初めに

μプロセッサ技術の進歩とネットワークの普及により、ビジネス分野も専用集中からUNIX(*1)を核にしたオープンシステムへと変化している。このようなオープンのマルチベンダ環境では更に高い価格性能比の実現が必須であり、そのためにはソフトウェアを含むシステムの基本性能を高めて行く必要がある。そこで、ソフトウェアからハードウェア資源への命令レベルのアクセス特性を調べ、更にソフトウェア自身の最適化(tuneup)を支援する手段として命令トレースシステムを開発した。

以下、本トレースシステムの動作原理と、実現したRISCアーキテクチャ(PA-RISC(*2)[1][2])での課題、適用例について報告する。

2. トレースの動作原理

本トレースシステムの動作原理を以下に示す。

(1) VM(仮想計算機)型の上位ソフトウェア構造

トレースは通常のKernelと同時に主記憶上にローディングされ、トレース開始の指示(openコール)で特権レベルを切り換え、Kernelを制御する上位ソフトウェアとなり、全割込みの制御権を握る。

(2) 分岐成功Trapの利用

トレースオーバーヘッド削減のため、命令毎に割込みを起こす方式でなく、分岐成功割込みを利用し、分岐から分岐までの命令をコピーする分岐間コピー方式。

(3) トレース出力専用ドライバ

バッファリングした命令データは外部記憶装置に蓄積し、専用の編集プログラムを用いて解析するが、この出力のためのI/OドライバはKernelのpureな動きを保持するためにKernelのドライバは使わず専用ドライバ化。

3. トレースシステムの構成

本トレースシステムは図1に示すようにKernel内に位置するStep Tracer部と通常のアプリケーションとして動く編集プログラム部からなる。

(1) 起動/終了部

特権レベルの変更と割込み制御部の切り換えを行う。

(2) 割込み制御(Int. Handler)部

すべての実割込みを受け付けデータ収集部に渡すと共に、Kernel依存の割込みはKernelに戻す処理を行う。

(3) データ収集(Data Collecting)部

2.(2)の命令コピーと共に割込みイベントやメモリアクセス情報等をコード化しバッファリングする。

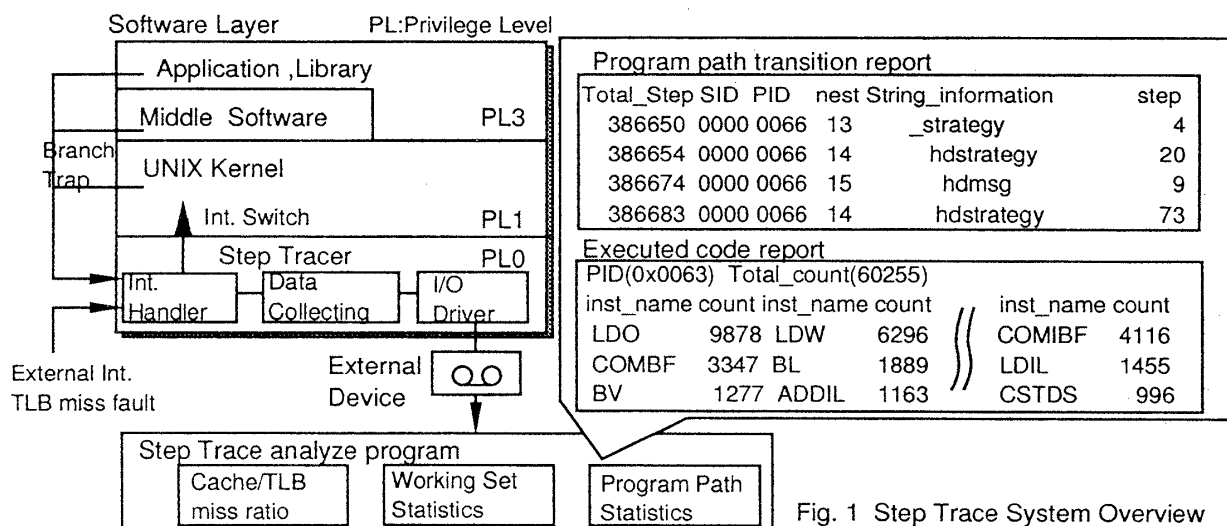


Fig. 1 Step Trace System Overview

Development of PA-RISC Step Trace System

Shigeo TAKASAKI, Masahiko NAKAHARA, Yutaka ENKO, Kimitosi YAMADA †, Yasufumi YOSHIZAWA

Systems Development Lab., Hitachi, Ltd. † Software Development Center, Hitachi, Ltd.

(*1) UNIX: UNIXオペレーティングシステム。UNIXはUNIX System Laboratories, Inc.が開発し、ライセンスしています。

(*2) PA-RISC: Precision Architecture - RISC. PA-RISCは米国Hewlett Packard社の商標です。

(4) I/Oドライバ

トレース情報を外部装置に出力する専用ドライバ。

(5) 編集プログラム

収集したトレース情報は以下の単位で編集する。

- ・関数フローと関数別のステップ数
- ・Instruction Mix
- ・プロセス参照ページ数(Working Set Size)
- ・Cache/TLBのミス率

4. 適用例

本トレースシステムの適用例としてDB(DataBase)処理環境でのキャッシュ/TLBの評価例を示す。

(1) 測定(トレース)条件

- ・マシン: 3050/R (自社RISCワークステーション)
- ・測定ジョブ: TPC-B(*3)ベンチマーク相当

(2) シミュレーション条件

- ・キャッシュ構成: I-/D-Cache, 256/256KB,
32byte Entry, Direct Map
- ・TLB: I-/D-TLB 96/96 Entry, Block TLB 4 Entry

(3) キャッシュ/TLBシミュレーション結果

表1 Cache/TLB Simulation Result

		1st Case	2nd Case
Cache	I-Cache miss率相対値	1.0	0.58
	D-Cache miss率相対値	1.0	0.26
TLB	I-TLB miss率相対値	1.0	1.0
	D-TLB miss率相対値	1.0	0.70

(注) ・1st case: cache/TLB共にクリアな状態でシミュレーションを開始(Worst case)

2nd case: 1st caseの終了状態で開始(Best case)

- ・各miss率はアクセス当たりであり、命令当りの場合にはLoad/Store命令比率をD-cache/D-TLB miss率に掛ける必要がある。

(4) 結果の検討

- ・上記結果は1stと2ndの比較であり、データ部の方が命令部に比較しlocalityが高いため、前の状態(1st)を利用でき、2ndのmiss率が下がる。
- ・I-TLBのmiss率が等しいのは、2nd caseでもほとんどTLBの内容を書き換えてしまうことを意味し、TLBエントリ数が少ないことも一因にある。

また図2は、Cache容量とMiss率の関係を本シミュレーションで求めたものである。

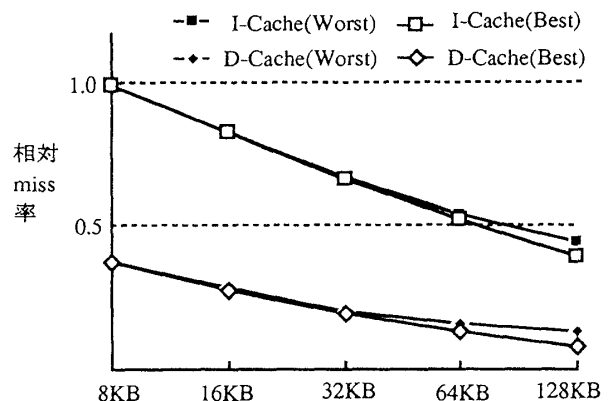


Fig.2 Cache Size Effect

本結果は単一プロセスのトレース結果でシミュレーションしたもので多重プロセス環境と異なるが、明示的なプロセススイッチ時にキャッシュクリアの処理を入れることで精度を上げることができる。

5. PA-RISC上の課題と対策

本トレースシステムの課題を対策を以下に示す。

(1) Nullification (NOP化命令) の扱い

<課題>分岐命令間コピーの方式のため、分岐命令間に次命令をNOPにするNullificationの検出が困難であり、検出コスト高(Overhead大)。

<対策>現在は実行命令としてカウント。

(2) トレースオーバーヘッド

<課題>分岐命令毎にトレーサに制御が移るが、それでも実モードに比べ500~1000倍遅くなる。この原因はトレーサ用ドライバの同期出力とトレーサ自身のキャッシュ占有、分岐命令頻度である。

<対策>トレース項目をオプション化し不要なトレースは止めることで対処。

6. 終わりに

本トレースシステムにより、PA-RISCマシン上で動くソフトウェアのマイクロな解析が可能となり、パスの最適化や特性解析に利用される。特に従来のUNIXで使われていたパス解析ツールのプロファイラと比較し再コンパイルの必要がなく、実行形式そのままで測定可能という特徴を持つ。

[参考文献]

- [1] Lee: Precision Architecture: IEEE Computer, 1989-1
- [2] DeLano他: A High Speed Superscalar PA-RISC Processor: IEEE Compcon'92

(*3): TPC(Transaction Processing Performance Council) - B: トランザクション処理性能評議会が設定されたベンチマークジョブでタイプBはデータベース中心