1G-04

処理時間に着目したLinuxカーネルの挙動解析

田原 秀晃† 芝 公仁†
†龍谷大学理工学部

1 はじめに

近年，あらゆるコンピュータシステムに対して，高い信頼性が求められるようになってきている．しかしシステムの信頼性は常に高い状態を維持しているわけではなく，システムの負荷が原因で処理効率が悪くなってしまう場合がある．このような処理効率の低下は再現性が低く，原因特定が困難とされている．なぜならパフォーマンス低下の原因となる要素が多岐にわたって存在するからである．そのため，負荷の原因が把握しづらい．Linuxの場合コード量が多く，システム全体の挙動を把握することが難しい．そのために処理効率低下時のLinuxの挙動を調査し解析するシステムの提案が必要である．

処理効率の低下などの現象が起こった場合にはログを確認したり，処理時間を調べる[1]ことが一般的である．本稿では処理効率低下の原因をLinuxカーネル内処理の処理時間に着目して解析を行う挙動解析システムについて提案する．

2 挙動解析システム

本章では，カーネル内処理の処理時間を解析する挙動解析システムについて述べる．図1に挙動解析システムの構成を示す．挙動解析システムは処理時間取得機構と処理時間解析機構の2つで構成される．挙動解析システムは，処理時間を計測する計算機をターゲットノード，処理時間を解析する計算機を制御ノードとした計算機を2台使用する．

処理時間取得機構はカーネル内処理の処理時間を計測する．挙動解析システムを1台の計算機で動作させた場合，処理時間解析機構のカーネル内処理の処理時間も計測する．その場合，Linuxカーネルの挙動に影響を与えてしまい処理時間取得機構の計測する処理時間が本来のカーネルのものではなくなってしまう．処理時間解析機構の計測の影響を少なくするために計算機を2台使用し，ターゲットノードでは計測に最低限必要なもののみを動作させる．その際に処理時間取得機構はターゲットノード内で動くが計算機本来の挙動を解析するために計測に影響を与えないようにする．

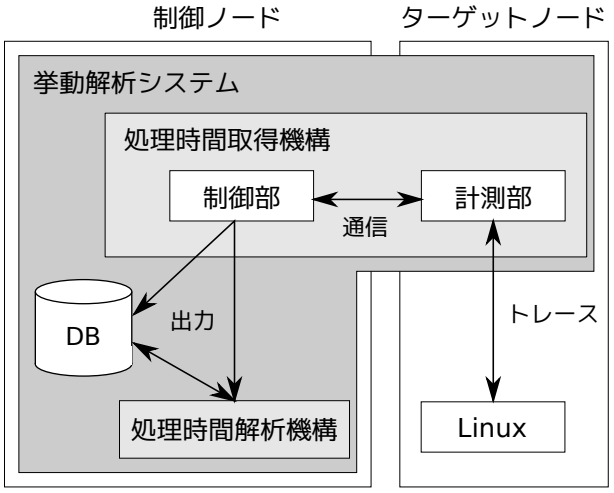

図1 挙動解析システムの構成

表1 処理時間データ

データ名	説明
処理名	計測対象となる関数名など
プロセスID	処理を実行したプロセスID
処理時間	処理にかかった時間

2.1 処理時間取得機構

本節では処理時間取得機構の動作について述べる．処理時間取得機構はターゲットノードで動作する計測部と制御ノードで動作する制御部の2つから構成される．

制御部は，計測部に実行を要求し，計測部から受け取った結果を処理時間解析機構やデータベースに出力する．計測部は制御部から要求を受け，Linuxカーネル内処理の処理時間を計測する．計測には，Linuxカーネルを動的にトレースすることができるSystemTapを使用してその挙動を観測する．

計測部が作成する処理時間データを表1に示す．処理時間解析機構が解析をするために，カーネル内処理の処理時間以外にも処理名とプロセスIDを取得する．処理時間データは計測部から制御部に送られ，処理時間取得機構とデータベースに出力する．

2.2 処理時間解析機構

本節では，前節で取得したカーネル内処理の処理時間の解析を行う処理時間解析機構について述べる．処

Behavior Analysis of the Linux Kernel by Processing Time
Hideaki Tahara†and Masahito Shiba†
†Faculty of Science and Technology, Ryukoku University

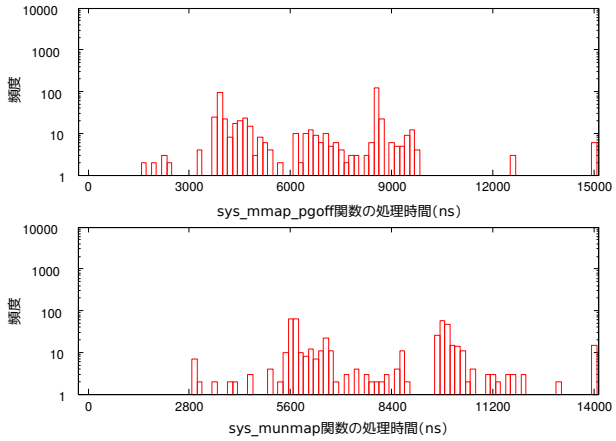

図 2　1 秒間に 1 回 Apache がリクエストを受けた時の関数の処理時間のヒストグラム

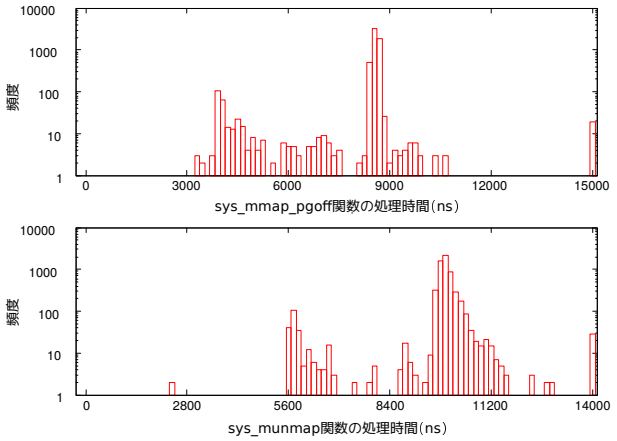

図 3　1 秒間に 5 回 Apache がリクエストを受けた時の関数の処理時間のヒストグラム

理時間解析機構は制御ノードで動作する．処理時間解析では処理時間の解析を行い，解析結果を表示，またはデータベースに格納する．

以下では処理時間の解析方法の 1 つとして横軸が秒数で縦軸が頻度を表したヒストグラムを用いる．処理時間取得機構の制御部から処理時間の結果を受け取る．受け取ったプロセス ID は処理時間を解析する際に，プロセス毎の処理時間の変化を解析するために用いる．整理した各処理時間のデータをヒストグラムにした際にヒストグラムによる適切な解析ができるようにビンの幅を決定する．ヒストグラムの縦軸は処理時間の頻度であり，どの横軸の区間で処理時間の頻度が多いのかを解析できる．処理時間は，頻度が少ないもので 0 回，多いもので 10,000 回以上と差があるため 1 つのヒストグラムで表示する際に縦軸の目盛りが細かくなるため頻度での解析が困難になる．その場合は，縦軸を対数にすることにより頻度が多い場合にもヒストグラムによる解析を可能にする．

2.3　処理時間解析機構の検討

ヒストグラムによる解析方法の検討をするために，sys_mmap_pgoff 関数と sys_munmap 関数の処理時間の解析を行った．これらの関数はファイルやデバイスをメモリにマップ，アンマップする関数である．

ターゲットノードでは Web サーバの Apache を動作させ，カーネルの sys_mmap_pgoff，sys_munmap 関数のトレースを行う．2 つの実験で処理時間解析を行いヒストグラムを作成した．一方は Apache が 1 秒間に 1 回リクエストを受け取った実験，もう一方は 1 秒間に 5 回リクエストを受け取った場合の実験である．2 つの実験による処理時間の解析結果を図 2，3 に示す．それぞれ，処理時間取得を 200 秒間行っている．ヒストグラムのビン 1 つの幅を sys_mmap_pgoff 関数は 150ns，sys_munmap 関数は 140ns とした．図 2 と図 3 を比較したところ，sys_mmap_pgoff 関数では 8,500ns あたり，sys_munmap 関数では 8,400ns から 11,200ns までの処理時間の頻度が多くなる傾向になった．また縦軸を対数にすることで頻度の回数の最小から最大までの差が大きい場合も，グラフの縦軸の値を増やす必要がなく，最小値，最大値の解析も同時に行えることを確認できた．

3　おわりに

本稿では処理時間に着目した Linux カーネルの挙動解析を目的とし，トレーサである SystemTap を用いたカーネル内処理の挙動解析システムを提案した．処理時間取得機構により取得した処理時間からヒストグラムを作成し，解析手法の有用性を検討した．

提案した挙動解析システムを Apache を動作させているターゲットノードを対象とし，プロセスが呼び出した sys_mmap_pgoff，sys_munmap 関数を対象として処理時間の解析を行った．処理時間解析機構によるヒストグラムの解析手法の有用性を検討した．横軸が秒数で縦軸が頻度にした際に頻度の多い場合でも縦軸を対数にすることによってヒストグラムによる解析を可能にした．

今後はヒストグラム表示による解析以外の様々な解析手法も検討し，処理効率が低下した際の Linux カーネルの挙動の変化を解析していく．

参考文献

[1] Nikolai Joukov, Avishay Traeger, Rakesh Iyer, Charles P.Wright, and Erez Zadok, ”Operating System Profiling via Latency Analysis” Proc. of USENIX Symp. on Operating Systems Design and Implementation (2006).