6J-4

EDF を基にしたスケジューリングアルゴリズムの比較分析

林　竜太†　　兪　明連†　　横山　孝典†

東京都市大学†

1. 研究背景

近年の技術の発展により組み込みシステムは高性能なプラットフォームを必要としてきており，マルチプロセッサ技術による性能の向上が望まれている．また，その中でもマルチプロセッサ環境下での効率の良いスケジューリングアルゴリズムが求められている．現在様々なスケジューリングアルゴリズムが提案されてきているが，スケジュール成功率が低い，オーバーヘッドが多いなどの問題点がある．今後新しいアルゴリズムを開発していくために，現在提案されているアルゴリズムの特徴を明確に捉えておくことが重要である．

2. 従来研究とその問題点

2.1 EDF

EDF(Earliest Deadline First)[1]は，デッドラインの早いジョブから順に高い優先度を付加し優先度の高いジョブから実行していく．時間の経過とともに優先度が変化する動的優先度スケジューリングアルゴリズムである．EDF はシングルプロセッサ環境下では最適とされているが，マルチプロセッサ環境下ではスケジュール成功率が低く，最適ではないことが知られている．スケジューラの起動はタスクの起動時，完了時に行われる．

2.2 EDZL

EDZL(Earliest Deadline Zero Laxity)[2]は，EDF にゼロ余裕時間ルールを適応したアルゴリズムである．ゼロ余裕時間ルールとは，余裕時間が 0 となったジョブに最高優先度を与えるものである．EDZL は通常は EDF と同じ動作をし，余裕時間が 0 のジョブが発生した場合にそのジョブに最高優先度を与える．余裕時間とはジョブの緊急度を表し，余裕時間が 0 であるとその時刻から休まず実行を続けないとデッドラインミスと起こしてしまう状態のことを表す．スケジューラの起動は毎単位時間行われる．

2.3 EDCL

EDCL(Earliest Deadline Critical Laxity)[3]は，通常は EDF と同じ動作をし，ジョブの起動時，完了時に余裕時間がクリティカル(危機的状況)になったジョブに最高優先度を与えるアルゴリズムである．「余裕時間がクリティカル」とは，あるジョブの余裕時間が，デッドラインの最も早いジョブの最小の残り実行時間よりも小さくなったことを指す．スケジューラの起動はタスクの起動時，完了時に行われる．

「Analysis of scheduling algorithm based on EDF」
†Ryuta Hayashi, Myungryun Yoo, Takanori Yokoyama
†Tokyo City University

2.4 問題点のまとめ

表 1 に従来研究の問題点をまとめる．
表 1 より，EDF を基にして提案された EDZL はスケジュール成功率が高いが余裕時間の計算によるオーバーヘッドが多く実装を困難にしている．また，EDCL はスケジューラの起動をタスクの起動時，完了時に制限したことによってオーバーヘッドは EDZL に比べ削減されたが，その分スケジュール成功率が低下してしまった．

表 1　従来研究の問題点．

EDF	実装は容易だが，スケジュール成功率が低い
EDZL	スケジュール成功率は高いが，毎単位時間 余裕時間の計算を行うためオーバーヘッドが多い
EDCL	EDZL と比べ，余裕時間の計算をタスクの起動時， 完了時に制限したことによりオーバーヘッドは削減 できたがスケジュール成功率が低下

3. 研究目的

今後 EDCL よりもスケジュール成功率が高く，実装が容易なスケジューリングアルゴリズムを開発するために，EDZL, EDCL のスケジューラの起動タイミングを変型させ，従来手法との比較，分析を行いスケジューラの起動頻度が性能にどのような影響を与えているのかを考察する．

4. 変型モデル

以下に変形させた EDZL，EDCL について示す．以下の 2 つのアルゴリズムも分析対象とすることで，同じアルゴリズムでもスケジューラの起動頻度によってどのような違いが発生するのかを考察する．

4.1 mEDZL

mEDZL(modified Earliest Deadline Zero Laxity)は，EDZL のスケジューラの起動をタスクの起動時，完了時に制限するアルゴリズムである．従来の EDZL と比べ，スケジューラの起動をタスクの起動時，完了時に制限したことによって優先度変更の条件は変わらないが，プリエンプション(割り込み)が発生するタイミングが減少しスケジュール成功率が低下すると考えられる．

4.2 mEDCL

mEDCL(modified Earliest Deadline Critical Laxity)は，EDCL のスケジューラの起動を毎単位時間行うアルゴリズムである．EDCL と比べ．スケジューラの起動を毎単位時間行うことで優先度変更の条件は変わらないが，プリエンプションが発生するタイミングが増加しスケジュール成功率が向上すると考えられる．

Copyright ©2014 Information Processing Society of Japan.
All Rights Reserved.

5. 評価方法

従来手法と変型モデルの 5 つのアルゴリズムについてシミュレーションを行う．評価項目はスケジュール成功率，デッドラインミス回数，デッドラインオーバー量，コンテキストスイッチ回数の 4 項目である．システム利用率が 30%から 100%までのタスクセットをそれぞれ 1000 個投入し，プロセッサ数は 4，8，16 個の 3 通りとした．

6. シミュレーション結果

図 1 にスケジュール成功率，図 2 にデッドラインミス回数，図 3 にデッドラインオーバー量，図 4 にコンテキストスイッチ回数のグラフをそれぞれ示す．なおプロセッサ数はいずれも 16 の場合である．

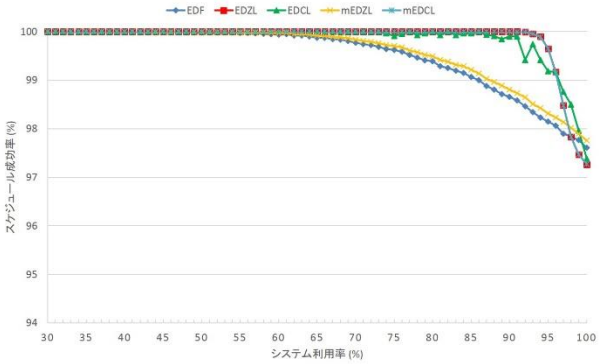

図 1　スケジュール成功率のグラフ

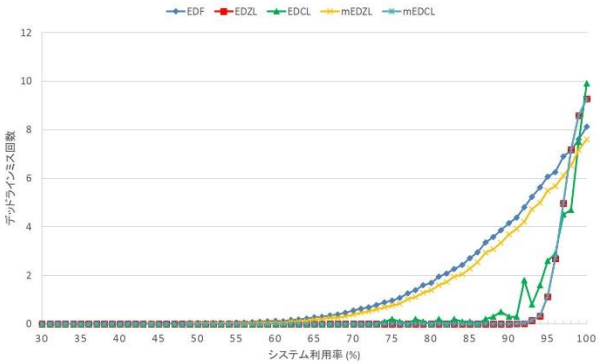

図 2　デッドラインミス回数のグラフ

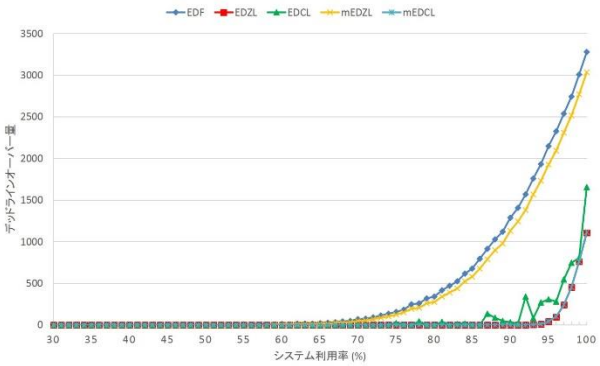

図 3 デッドラインオーバー量のグラフ

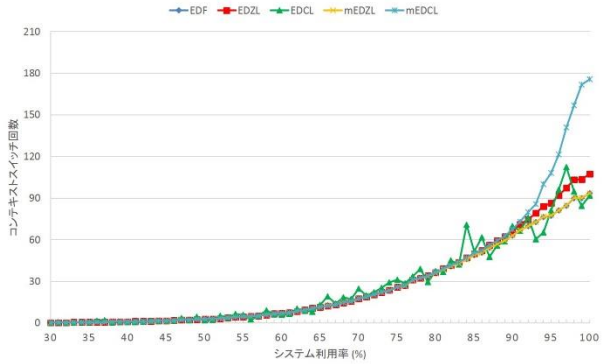

図 4　コンテキストスイッチ回数のグラフ

7. 考察

図 1 より，スケジューラを毎単位時間起動する EDZL，mEDCL はスケジュール成功率が高く，スケジューラの起動がタスクの起動時，完了時のみの EDF，mEDZL はスケジュール成功率が低い．EDCL のスケジューラの起動は EDF，mEDZL と同じだがスケジュール成功率が EDF，mEDZL と比べると高い．しかし，システム利用率によって値が上下している．これは与えられたタスクセットに依存していると考えられる．

また，EDZL，EDCL，mEDCL はシステム利用率が 100%に近づくにつれて EDF，mEDZL よりもスケジュール成功率，デッドラインミス回数の値が悪くなっている．利用率が高い高負荷な状態では EDZL，EDCL の優先度変更の条件よりも，EDF によってスケジュールされたタスクの数の方が上回っているということになる．

mEDCL を例に考えると，毎単位時間スケジューラを起動させてもスケジュール成功率などが EDZL よりも上回ることはなく，ほぼ同じ結果となった．図 4 の結果からも，mEDCL の方が大量にコンテキストスイッチが発生してしまった．このことから，アルゴリズムによって適切なスケジューラの起動タイミングがあることがわかる．

スケジュール成功率が高く実装が容易なアルゴリズムを今後開発していくためには，スケジューラの起動頻度をなるべく抑えつつ予測性の高いアルゴリズムにしなければならない． また，高負荷な状態でも高いスケジュール成功率を維持しつつタスクセットに依存しないどのようなタスクにも対応できる安定したアルゴリズムを目指す必要がある．

謝辞

本研究の一部は，JSPS 科研費 24500046 の助成を受けたものです．

参考文献

1) C. L. Liu and J. W. Layland, “Scheduling Algorithms for Multiprogramming in a Hard-Real-Time Environment”, Journal of ACM, Vol. 20, No. 1, pp. 46-67 (1973)
2) S. Cho, S. Lee, A. Han, and K. Lin, “Efficient real-time scheduling algorithms for multiprocessor systems”, IEICE Trans Communications, E85-B(12), pp. 2859-2867 (2002)
3) S, Kato and N, Yamasaki, “Global EDF-based scheduling with laxity-driven priority promotion”, Journal of systems Architecture, Vol. 57, pp. 498-517 (2011)

Copyright ©2014 Information Processing Society of Japan.
All Rights Reserved.