4K-02

クリティカルの基準変更による動的スケジューリングアルゴリズムの性能変化の分析

佐々木　理成†　　兪　明連†　　横山　孝典†
東京都市大学†

1. 研究背景

近年，組込みリアルタイムシステムではマルチプロセッサ技術の利用が一般化してきている．しかし，従来のシングルプロセッサ環境下で動的優先度として最適なアルゴリズムであるEDF(Earliest Deadline First)[1]はマルチプロセッサ環境下では最適ではないことが知られている．現在，EDF を基にしたマルチプロセッサ向けのアルゴリズムはいくつか提案されているが，確立には至っていない．本研究では，EDF にクリティカルタスクルールを付加したEDCL(Earliest Deadline Critical Laxity)[2]の優先度判定方法であるクリティカルの基準を変更し，その性能評価結果から適切なクリティカルの基準について議論する．

2. システムモデル

以下にシステムモデルを示す．システムはM個のプロセッサを持ち，N個の周期的タスクから構成されるタスクセット$\tau = \{\tau_1, \tau_2, \cdots, \tau_N\}$が与えられる．各タスク$\tau_i$は$\tau_i = (C_i, T_i)$と定義され，$C_i$，$T_i$はそれぞれ$\tau_i$の最悪実行時間と周期を表す．また，$U_i = C_i/T_i$をタスク$\tau_i$の利用率とし，$U(\tau)/M$ をシステム利用率と定義する．各タスクτ_iは周期T_iの間隔でジョブを生成し，タスクτ_i が生成するｋ番目のジョブをτ_{ik}と表す．τ_{ik}のリリース時刻をr_{ik}とし，デッドラインをd_{ik}とするとき，d_{ik}は次のジョブのリリース時刻と一致する．ある時刻tにおいて，ジョブτ_{ik}の残り実行時間を$C_{ik}(t)$とするとき，τ_{ik}の余裕時間 $L_{ik}(t)$を$L_{ik}(t) = d_{ik} - t - C_{ik}(t)$と定義する．

A study on dynamic scheduling algorithm by critical criteria
† Risei Sasaki, Myungryun Yoo, Takanori Yokoyama
‡ Tokyo City University

3. 従来研究

EDF(Earliest Deadline First)

EDF はデッドラインの早いタスクに高い優先度を与えるアルゴリズムである．マルチプロセッサ環境下ではスケジュール成功率が低いという問題点がある．スケジューラの起動はタスクの起動時と完了時のみ行う．

EDCL(Earliest Deadline Critical Laxity)

EDCL は EDF においてクリティカルなタスクに最高優先度を与えるアルゴリズムである．クリティカルなタスクとは，EDF でスケ ジュールする場合に実行されるタスクの中で最小の残り実行時間を求め，その値よりも小さい余裕時間を持つタスクのことをいう．スケジューラの起動はタスクの起動時と完了時のみ行う．EDF に比べスケジュール成功率は向上したが，未だに低いという問題点がある．

4. 分析モデル

以下に EDCL におけるクリティカルの基準を変更した 6 つのパターンを示す．任意のスケジューラの起動時刻t_sにおいて，EDF でソートした場合に実行されるタスクの中で最小の残り実行時間を $e_{min}(t_s)$とし，タスクτ_iの残り実行時間を$C_i(t_s)$，余裕時間を$L_i(t_s)$とする．

- **パターン 1**: $e_{min}(t_s) > L_i(t_s)$（元の EDCL）
- **パターン 2**: $\lceil e_{min}(t_s) \times 1/2 \rceil > L_i(t_s)$
- **パターン 3**: $\lceil e_{min}(t_s) \times 3/2 \rceil > L_i(t_s)$
- **パターン 4**: $\lceil C_i(t_s) \times 1/2 \rceil > L_i(t_s)$
- **パターン 5**: $\lceil C_i(t_s) \times 1/4 \rceil > L_i(t_s)$
- **パターン 6**: $\lceil C_i(t_s) \times 3/4 \rceil > L_i(t_s)$

パターン 2 と 3 はそれぞれ元のパターン 1 に比べ，クリティカルになりにくく，またはなりやすく設定する．パターン 4 以降は$e_{min}(t_s)$ではなく$C_i(t_s)$を用いて周期の長いタスクでも比較的余裕をもって実行されやすいようにしている．

5. 評価

パターン 1～6 のアルゴリズムについてシミュレーションにより評価を行う．システム利用率 70%から 100%の範囲で 5%おきに一様乱数を用いて生成したタスクセットをそれぞれ1000 個投入し，各アルゴリズムでシミュレーションをする．プロセッサ数は 4，8，16 の 3 通りとし，各タスク利用率の範囲は[0.01，1.0]とする．評価項目はスケジュール成功率，コンテキストスイッチ回数の平均，スケジューラ起動回数の平均，デッドラインオーバー量の平均の 4 つである．図 1～図 4 にプログラム数が 16 の場合のそれぞれのグラフを示す．

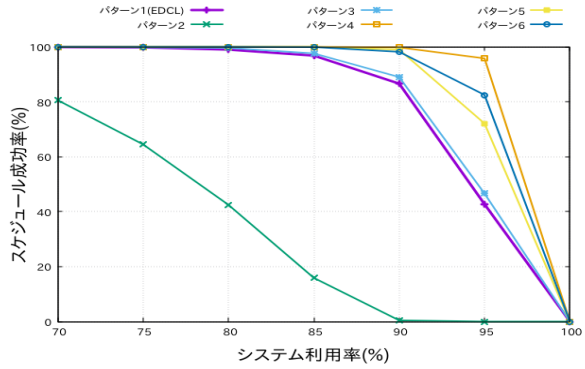

図 1. スケジュール成功率

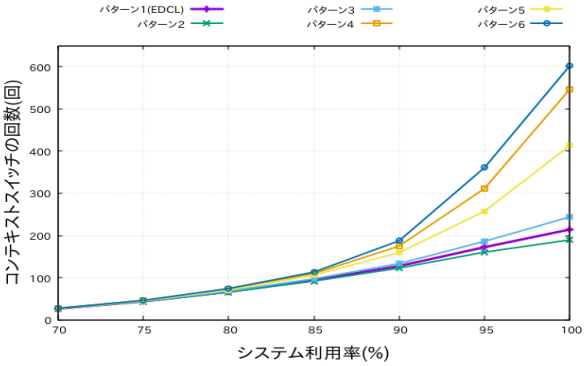

図 2. コンテキストスイッチ回数の平均

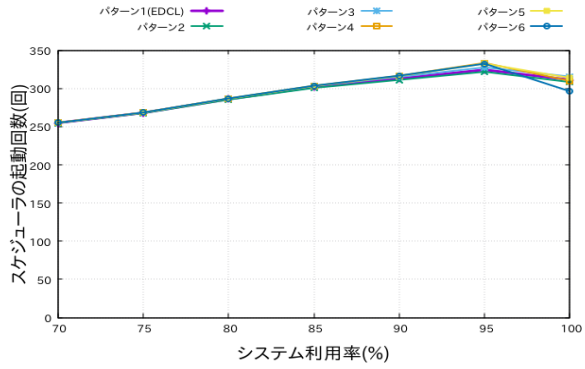

図 3. スケジューラ起動回数の平均

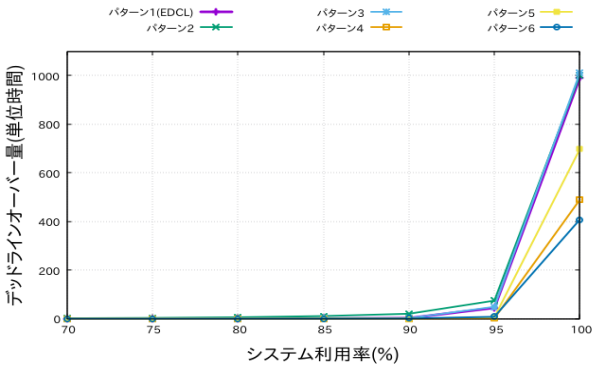

図 4. デッドラインオーバー量の平均

図 1 より，$e_{min}(t_s)$を用いるパターン 1～3 より $C_i(t_s)$を用いるパターン 4～6 の方がスケジュール成功率は高くなった．これは，周期の長いタスクが高負荷な状態でもクリティカルになりやすいためだと考えられる．また，スケジュール成功率の上昇に伴い，図 4 のデッドラインオーバー量の平均もパターン 4～6 では抑えることに成功した．一方で，図 2 のコンテキストスイッチ回数については，比較的クリティカルになりやすいパターン 4～6 で優先度の変更が頻発した．図 3 のスケジューラ起動回数は，全てのパターンでタスクの起動時と完了時のみにしていたので大差はなかった．スケジュール成功率の面では，最も高いパターン 4 がクリティカルの基準に適している．パターン 4 はデッドラインオーバー量も抑えられているが，コンテキストスイッチの回数に問題がある．

6. 結論

本研究では，EDCL の優先度判定方法であるクリティカルの基準を変更し適切なクリティカルの基準について議論した．今後は分析したモデルのスケジュール可能性解析を行う予定である．

謝辞

本研究は JSPS 科研費 20K11755 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 the ACM, Vol. 20, No. 1, pp. 46-67, 1973.

[2] S. Kato and N. Yamazaki: “Global EDF-based scheduling with laxity-driven priority promotion”, Journal of systems Architecture, Vol. 57, No. 5, pp. 498-517, 2011.