数理モデル化と問題解決 33－7
（2001．3．15）

VBRストリーム処理のための適応的スケジューリングポリシとその性能評価

滝沢泰久† 芝 公仁†† 大久保英嗣†††

動画や音声などの連続メディアを扱うストリーム処理タスクをスケジュールする場合，それらの時間制約を満たすために，最悪処理時間に基づいたリアルタイムスケジューラが利用されてきた．しかし，多くの連続メディアデータは，圧縮や差分処理などにより，処理時間は必ずしも一定とはならない．このような連続メディアデータを従来の最悪処理時間に基づいた方式で処理すると，過度なCPU利用率の予約のために，システム全体のCPU利用率が大きく低下する．本稿では，以上の問題を解決するために，ストリーム処理タスクの時間制約と処理量の変動に動的に適応するスケジューリングポリシを提案し，さらに，その性能評価について述べる．

An Adaptive Scheduling Policy for VBR Stream Processing and Its Performance Evaluation

YASUHISA TAKIZAWA,† MASAHITO SHIBA†† and EIJI OKUBO†††

Real-time scheduling policies based on the worst-case execution time have been used for stream processing tasks which manipulate continuous media such as audio and video. However, the processing time for continuous media stream dynamically changes as the result of compressions and differences between data. Therefore, conventional schemes cause excessive reservations of the processing time since the processing time of each continuous datum is shorter than its worst-case execution time. In this paper, a new scheduling policy which is adaptable for stream processing tasks with timing constraints and processing delay is proposed, and described about its performance evaluation.

1. はじめに

動画や音声などの連続メディアを処理するアプリケーションが数多く出現している．連続メディアアプリケーションは，処理するメディアの特性上，周期タスクモデルに基づいた時間制約を持つ．この時間制約を満たすために，最悪実行時間と周期タスクモデルに基づいたリアルタイムスケジューラが利用されてきた．

一方，連続メディアデータは，入出力装置から固定周期で生成／消費され，そのデータ量は圧縮や差分処理により可変量となる．すなわち，連続メディアデータは，Variable Bit Rate（以降VBR）データであると言える．このような連続メディアデータのストリーム処理は，パイプラインモデルにより行われる．パイプライン上のタスクにおいて，VBRデータの処理時間は可変となるため，後続のタスクにおけるデータの到着は周期性を失う．従って，周期タスクによりストリーム処理を行った場合，タスクにデータ未到着による待ち時間が発生しスケジュール可能性が低下する．これを回避するためには，パイプライン上の隣接するタスク間に十分な初期位相が必要となり，エンドーエンドの処理遅延が大きくなる．また，最悪実行時間に基づきCPU利用率を予約することは過度な予約量となり，システム全体でのCPU利用率が低下する．

本稿では，以上の問題点を解決するために，

- 最悪実行時間より短い時間をCPU使用時間とする．
- タスクの処理時間は，ランダムに変動する．
- タスクはパイプラインモデルにより通信する．
- 連続メディアデータは，その入出力装置からVBR／固定周期で生成／消費される．
- 複数のストリーム処理が混在する．

を前提として，従来の方式より高いスケジュール可能性を導く適応的スケジューリングポリシを提案する．さらに，その性能評価についても述べる．

提案ポリシでは，ストリーム処理モデルを連続メディ

† （株）ATR環境適応通信研究所
ATR Adaptive Communications Research Laboratories
†† 立命館大学大学院理工学研究科
Graduate School of Science and Engineering, Ritsumeikan Univ.
††† 立命館大学理工学部情報学科
Department of Computer Science, Facutly of Science and Engineering, Ritsumeikan Univ.

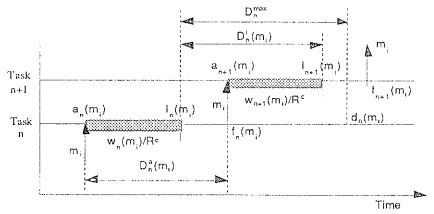

図 1　LBAP における時間属性
Fig. 1　Timing attributes on LBAP.

ア資源モデル Linear Bounded Arrival Process[3]（以降 LBAP）に VBR データ処理のための変更を加えたモデルとし，Parallel Distributed Processing モデル（以降 PDP モデル）[2] と熱力学的モデル [1] を動作メカニズムとしている．これにより，VBR データに動的に適応するタスクスケジューリングが可能となる．

2. VBR ストリーム処理モデル

2.1 LBAP に基づいた従来の処理モデル

従来の方式[4]では，ストリームデータを一定のメッセージの到着レートと最悪実行時間により特徴付け，CPU 利用率を予約している．従って，従来の方式は，最悪実行時間に基づいた Constant Bit Rate（以降 CBR）ストリーム処理として考えられ，LBAP に基づき，次のよう定義される（図 1 参照）．

R^c: constant message rate (messages/second)
W^{max}: maximum workahead (messages)
C^{max}: maximum processing time for a message (seconds/message)

$$
\begin{aligned}
&W_n^{max} = \max_i(w_n(m_i)) \\
&w_n(m_0) = 0 \\
&w_n(m_i) = max(0, w_n(m_{i-1}) \\
&\qquad - (a_n(m_i) - a_n(m_{i-1}))R^c + 1) \\
&l_n(m_i) = a_n(m_i) + w_n(m_i)/R^c \\
&D_n^l(m_i) = l_{n+1}(m_i) - l_n(m_i) \\
&d_n(m_i) = l_n(m_i) + D_n^{max} \\
&D_n^{max} = \max_i(D_n^l(m_i)) \\
&D_n^a(m_i) = a_{n+1}(m_i) - a_n(m_i)
\end{aligned}
$$

ただし，タスク n の i 番目のメッセージにおいて，$w_n(m_i)$ はメッセージ到着時の未処理メッセージ数，$a_n(m_i)$ はメッセージ到着時刻，$l_n(m_i)$ はメッセージの論理到着時刻，$D^a(m_i)$ は処理遅延時間，$D^l(m_i)$ は論理処理遅延時間，$d_n(m_i)$ はデッドライン時刻である．

上記の定義された時間属性において，予約 CPU 利用率が $C^{max}R^c$ の場合，出力装置の時間制約を満たすには，タスク間に，次のような初期位相が必要である．

$$
a_n(m_i) \geq a_0(m_0) + \sum_{j=0}^{n-1}(W_j^{max} + 1)/R^c
$$

以上のことから，出力装置の時間制約を保証する場合，過度な CPU 予約量と大きな初期位相が必要となる．

2.2 VBR データにおけるストリーム処理モデル

本節では，最悪実行時間に基づく CBR ストリーム処理モデルの問題を解決するため，メッセージ処理時間を最悪実行時間 C^{max} より短い任意の時間 C^{req} とする処理モデルについて述べる．この処理モデルを，LBAP を用いて，次のように定義する．

R^c: constant message rate (messages/second)
b: actual workahead messages (message)
C^{req}: request processing time for a message (seconds/message)

上記パラメータで CPU 利用率を予約している場合，すべてのメッセージ処理時間が $1/R^c$ 以下ではない．従って，2.1 節の論理時刻を適用できないので，次のように属性を変更する．

$w_n(m_i)$ の代わりに，タスク n の時刻 t に実測された未処理メッセージ数を $b_n(t)$ を用いる．これを実効未処理メッセージ数と呼ぶ．また，$l_n(m_i)$ に代わりに，タスク n の i 番目のメッセージが初めて処理対象となった実際の時刻 $e_n(m_i)$ を用いる．この時刻をメッセージ実効到着時刻とし，次のように定義する．

$$
\begin{aligned}
&e_n(m_0) = a_n(m_0) \\
&e_n(m_i) = max(a_n(m_i), f_n(m_{i-1}))
\end{aligned}
\qquad (1)
$$

ただし，$f_n(m_{i-1})$ は，タスク n の $i-1$ 番目のメッセージの処理完了時刻である．

さらに，デッドライン時刻を次のように定義する．

$$
d_n(m_i) = e_n(m_i) + b_{n+1}(e_n(m_i))/R^c \qquad (2)
$$

その他の定義は，2.1 節と同様である．

2.3 VBR ストリーム処理モデルにおける特性

本節では，VBR ストリーム処理モデルに関する特性について述べる．

[特性 1] 予約 CPU 利用率が $C^{req}R^c$ の場合，処理完了時刻は，以下の条件で，デッドライン時刻を満たす．

$$
b_{n+1}(e_n(m_i))/R^c \geq D_n^a(m_i)
$$

[特性 2] 特性 1 の条件を満たす場合，処理に必要な CPU 利用率は，以下の条件を満たすならば，予約 CPU 利用率を超えない．

$$
b_{n+1}(e_n(m_i)) \geq C_n(m_i)/C_n^{req} \qquad (3)
$$

ただし，$C_n(m_i)$ は，タスク n の i 番目のメッセージ処理時間である．

[特性 3] 予約 CPU 利用率が $C^{req}R^c$ の場合，初期位

相が以下を満たすならば，出力装置の時間制約が満たされる．

$$a_n(m_0) \geq a_0(m_0) + \sum_{j=0}^{n-1} D_j^a(m_i) \qquad (4)$$

3. 提案スケジューリングポリシ

3.1 スケジュール可能性を高める制御

特性3の条件を満たすには，タスク間に十分な初期位相が必要となり，エンドーエンド処理遅延時間が増す．本節では，少ない初期位相で，出力装置の時間制約を満たす可能性を高める制御について考える．

式（4）に，パイプラインモデルにおけるメッセージ到着時刻 $a_n(m_i)$ と処理完了時刻 $f_{n-1}(m_i)$ が一致すること，およびタスク0のメッセージの到着時刻は周期的であることを適用すると，次式が得られる．

$$\sum_{k=0}^{i}\{\sum_{j=0}^{n-1}(D_j^a(m_{k+1}) - D_{j+1}^a(m_k)) + (D_{in}^a(m_{k+1}) - D_0^a(m_k)) + (D_n^a(m_{k+1}) - D_{out}^a(m_k))\} \leq 0$$

ただし，$D_{out}^a(m_k)$ は出力装置の k 番目のメッセージ処理遅延時間，$D_{in}^a(m_{k+1})$ 入力装置の $k+1$ 番目のメッセージ処理遅延時間である．

上式から，隣接するタスク間のメッセージ処理遅延時間を均等にすることで，出力装置の時間制約を満たすことができる．従って，次のような制御を行う．

- 同一パイプラインのタスク優先度を連動させる．
- タスク n の $b_n(e_n(m_i))$ と後続タスク $n+1$ の $b_{n+1}(e_n(m_i))$ を比較し，前者の値が大きい場合はタスク n の優先度を高める．また，後者の値が大きい場合は優先度を低める．
- 実測された処理遅延時間が特性1の条件を満たさないならば，優先度を高める．
- 実測された処理遅延時間が特性2の条件を満たさないならば，優先度を低める．

以上により，出力装置の時間制約を満たし，かつ，エンドーエンドの処理遅延時間を小さくできる．

3.2 適応デッドライン時刻

ストリーム処理タスクは，ソフトリアルタイムタスクとして考え，デッドライン時刻に許容遅延幅 h_n を持たせる．デッドライン時刻は，式（2）で求められる時刻に，許容遅延幅内 $[-h_n, h_n]$ で，スケジュール可能性を高める制御による修正を行った時刻とする．この時刻を適応デッドライン時刻 $d_n^{adap}(m_i)$ と呼ぶ．

3.3 複数のストリーム処理と多重制約

マルチメディア環境では，複数のストリーム処理を扱うシステムが多くなると予想される．複数のストリーム処理に対して3.1節の制御を行う問題は，同時に制約を満たす状態を探し出す多重制約問題である．

提案ポリシは，この多重制約問題を処理するために，PDPモデルを応用し，タスク間の依存関係をPDPモデルの相互結合ネットワーク（以降，ネットワーク）に次のように対応づける．

- 各ユニットの状態値は，最高値（1）を $d_n^{adap}(m_i)$ の最短時刻 $d_n(m_i) - h_n$ に，最低値（0）を $d_n^{adap}(m_i)$ の最長時刻 $d_n(m_i) + h_n$ に対応づける．
- ユニット間の結合は，通信するタスク間の場合正の重みを，通信しないタスク間の場合負の重みを持たせる．
- 各ユニットに与えられる外部条件は，各タスクの変動する処理遅延時間に応じて，3.1節で述べた制御とする．

以上により，複数のパイプラインのタスク群における多重制約を処理する．しかし，PDPモデルのネットワークは，初期状態に依存してある制約充足度の高い状態に至り，動作は静止する．一方，ストリーム処理環境において，各タスクの処理遅延時間は常に変化する．ネットワークは，この変化に応じて，1つの状態に静止することなく，いくつかの制約充足度の高い状態間を移動する必要がある．このため，PDPモデルに熱力学的モデル[1]を加える．熱力学的モデルは，温度による物質の熱揺動をPDPモデルの処理において擬似する．すなわち，高い充足状態から離れた場合は温度を高くし，ネットワーク動作を大きく振動させ新しい状態を見つける可能性を高める．一方，高い充足状態の近傍にある場合は温度を低くし，ネットワーク動作を現在の状態の近傍で小さく振動させる．この温度による熱揺動制御により，ネットワークは各タスクの変動する処理遅延時間に連続的に動作する．

4. 性能評価

提案ポリシを，Solelc[5]上に実装し，性能評価を行った．本章では，その結果について述べる．

4.1 評価方法と評価タスクセット

評価タスクセットを，次のタスクにより構成される3組のタスク群と，ランダムな負荷を生成する周期タスクより構成する．

- 周期的にメッセージを生成する入力装置
- 周期的にメッセージを消費する出力装置
- アプリケーションタスク，入力および出力サーバタスク

評価方法は，周期タスクモデルの方式，LBAPの方

表 1　評価タスクセットの時間属性
Table 1　Timing attributes for example taskset.

	$1/R^c$	h	C^{max}	C^{min}	C^{ave}
task1	$100ms$	$10ms$	$12ms$	$2ms$	$7ms$
task2	$100ms$	$10ms$	$28ms$	$2ms$	$15ms$
task3	$100ms$	$10ms$	$13ms$	$3ms$	$8ms$
task4	$70ms$	$10ms$	$8ms$	$2ms$	$5ms$
task5	$70ms$	$10ms$	$20ms$	$2ms$	$11ms$
task6	$70ms$	$10ms$	$8ms$	$2ms$	$5ms$
task7	$40ms$	$10ms$	$5ms$	$1ms$	$3ms$
task8	$40ms$	$10ms$	$10ms$	$2ms$	$6ms$
task9	$40ms$	$10ms$	$5ms$	$1ms$	$3ms$
task10	$30ms$	–	–	–	–

h: 遅延許容幅　C^{max}: 最大実行時間
C^{min} : 最小実行時間　C^{ave}: 平均実行時間

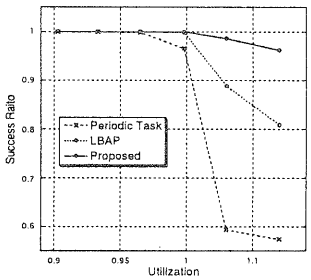

図 2　負荷状況に応じたスケジューリング成功率
Fig. 2　Scheduling success raito by utilization.

式および提案ポリシの 3 方式を，上記タスクセットのスケジューリング成功率で比較する．各タスクの時間属性を以下に示す．ただし，タスクの実行時間は，最大実行時間と最小実行時間の間をランダムに変動させた．

4.2　スケジューリング成功率

スケジューリング成功率 SR を，次のように定義する．

$$SR = \frac{Success(Load) + Success(Msg)}{Fin(Load) + Fin(Msg)}$$

$Success(Load)$ はデッドライン時刻までに処理を完了した負荷周期タスク数，$Success(Msg)$ は出力装置の処理開始時刻までに出力装置に到着したメッセージ数，$Fin(Load)$ は処理完了した負荷周期タスク数，$Fin(Msg)$ は出力装置に到着したメッセージ数である．

負荷状況に応じたスケジューリング成功率を図 **2** に示す．各タスク間の初期位相は，各タスクの 1 周期分の時間とした．また，LBAP の方式の D_n^{max} は初期位相と同じ時間とした．

図 2 から分かるように，提案ポリシは他の方式よりスケジューリング成功率が常に高い．また，過負荷の状況においても，高いスケジューリング成功率となる．

次に，初期位相に応じたスケジューリング成功率を図 **3** に示す．負荷は 1.03 とし，初期位相は各タスクの周期の倍数で設定した．

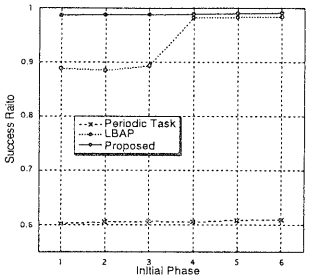

図 3　初期位相状況に応じたスケジューリング成功率
Fig. 3　Scheduling success raito by initial phase.

図 3 から分かるように，提案ポリシは，少ない初期位相でスケジューリング成功率が改善する．

以上のことから，提案ポリシは，従来の方式と比較して，次のような特性をもつ．

- スケジューリング可能性が高く，高負荷時にその差が著しい．従って，CPU 利用率を高くできる．
- 少ない初期位相でスケジューリング成功率が改善される．従って，少ないエンドーエンド処理遅延時間でスケジューリングできる．

5.　おわりに

本稿では，VBR ストリーム処理タスクのスケジューリング方式において，高い CPU 利用率環境でも，連続メディア出力装置の時間制約を満たし，かつ，エンドーエンドの処理遅延時間を小さくするポリシを提案した．さらに，その性能評価を行い，その有効性とを示した．

参　考　文　献

1) R. Yavatkar and K. Lakshman: Optimization by Simulated Anneling, *Science*, Vol. 220, pp. 671–680 (1983).
2) D. E. Rumelhart, J. L. McCelland and the PDP Research Group: *PARALLEL DISTRIBUTED PROCESSING*, The MIT Press (1986).
3) D. P. Anderson: Metascheduling for continous media, *Trans Comput Syst ACM*, No. 11, pp. 226–252 (1993).
4) R. Govindan and D. P. Anderson: Scheduling and IPC mechanisms for continuous media, *Proceeding of the 13th ACM on Operating Systems Principles*, pp. 68–80 (1991).
5) 芝公仁，大久保英嗣:分散オペレーティングシステム Solelc の構成，情報処理学会研究会報告 *2000-OS-84*, pp.237-244 (2000).