7U－2

Rate-Monotonic アルゴリズムを用いた分散リアルタイムスケジューラの検討

田辺 雅則　遠城 秀和
NTT データ通信(株)　NTT データ通信(株)

1　はじめに

分散環境におけるトランザクション処理においても、リアルタイム処理を保証することが要求される。分散環境におけるリアルタイム処理では、リアルタイム処理を保証する計算機に処理を分散させる分散リアルタイムスケジューリングが重要である。

分散環境でリアルタイム処理を保証する場合、計算機内のスケジュールを行うローカル・スケジューラと計算機間の負荷分散を目的としたスケジュールを行うグローバル・スケジューラを合わせもつことが多い[1]。しかし、グローバル・スケジューラが負荷分散のみを考慮して処理を分散する場合、リアルタイム処理を保証できる計算機に分散されるとは限らない。

本報告では、シングル環境でリアルタイム処理を保証するために使用される Rate-Monotonic アルゴリズムを拡張したものを用いて分散環境でリアルタイム処理を保証する計算機を決定し、メッセージベースの処理依頼方式を用いて処理を分散させるグローバル・スケジューラの実現方式について述べる。

2　分散リアルタイムスケジューラの構成

2.1　分散リアルタイムスケジューラの構成方法

多くの場合、分散環境の構成は動的に変更され、各計算機の計算能力は異なる。そのため、分散環境の構成や各計算機の計算能力や負荷を考慮して、分散リアルタイムスケジュールを行う必要がある。

そこで、分散環境の状態を管理して計算機間で処理のリアルタイムスケジュールを行うグローバル・スケジューラと、各計算機で処理のリアルタイムスケジュールを行うローカル・スケジューラで分散リアルタイムスケジューラを構成する。この場合、ローカル・スケジューラは分散環境の状態や他の計算機の負荷管理、他の計算機へ処理の動的な再分散をする必要がなくなるため、ローカル・スケジューラの計算コストや通信コストを下げることができる。また、このように構成すると、ローカル・スケジューラにリアルタイムスケジュールを行う既存の OS のスケジューラを利用できる。

Distributed Real-time Scheduler using Rate-Monotonic Algorithm
Masanori TANABE & Hidekazu ENJO
NTTDATA COMMUNICATION SYSTEMS CORPORATION

2.2　グローバル・スケジューラにおける問題

グローバル・スケジューラで負荷分散を行い、ローカル・スケジューラでリアルタイム処理を保証するスケジューリング方式では、リアルタイム処理を保証できなかった場合に、グローバル・スケジューラが処理の再分散を行うことになり不効率である。そこで、グローバル・スケジューラは、リアルタイム処理を保証する計算機に処理を分散させるための機能も持つ必要がある。

3　分散リアルタイムスケジューラの実現

3.1　分散リアルタイムスケジューラの実現方法

実現した分散リアルタイムスケジューラでは、グローバル・スケジューラにおいてリアルタイム処理を保証する計算機を決定し、決定された計算機で動作するローカル・スケジューラに処理を依頼する方式を用いた。この方式では、グローバル・スケジューラには、リアルタイム処理を行う機能と処理を分散させる機能が必要である。そこで、これらの機能を以下の2つのプログラムで実現した。

(1) スケジュール処理プログラム　リアルタイム処理を保証する計算機の算出と、それを保証する計算機に処理を分散させる。

(2) 処理実行プログラム　各計算機で動作し、スケジュール処理プログラムによって分散させられた処理をローカル・スケジューラに依頼する。

図1に実現した分散リアルタイムスケジューラの構成図を示す。

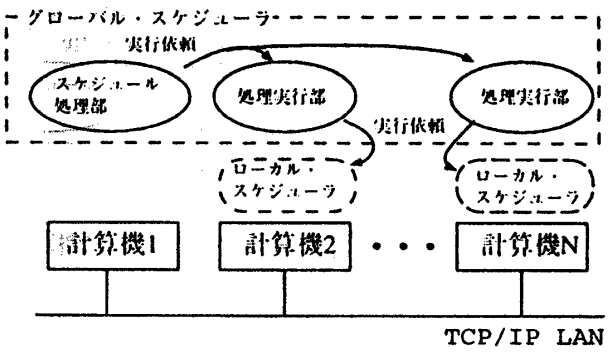

図 1: 分散リアルタイムスケジューラの構成

3.2 スケジュール処理プログラム

3.2.1 リアルタイム処理を保証する上での問題

グローバル・スケジューラには、リアルタイム処理を保証する計算機に処理を分散させるため、処理を実行する前にリアルタイム処理の可能性を計算する Rate-Monotonic アルゴリズム [2] を用いる。

Rate-Monotonic アルゴリズムは、周期的タスクのうち周期の短いタスクの優先順位を高くし、プリエンプティブなスケジューリングをシングル環境で行うリアルタイム・スケジューリングアルゴリズムである。この Rate-Monotonic アルゴリズムを分散環境で使用する場合、以下の拡張が必要である。

(1) 処理を実行する計算機を表す。
(2) 各計算機の処理能力の差を表す。

3.2.2 分散環境を考慮した拡張 Rate-Monotonic アルゴリズム

(1) Rate-Monotonic アルゴリズム

シングル環境を前提とした Rate-Monotonic アルゴリズムの計算式を式 (1) から式 (4) に示す。

$$W_i(t) = \sum_{j=1}^{i} C_j \lceil t/T_j \rceil \qquad (1)$$

j :処理番号　　i :処理の個数
C_j:処理jの計算時間　　T_j:処理jの周期
t :計算を行う時間区間
ただし、$t = max_{\{1<j<i\}} T_j$である。

$$L_i(t) = W_i(t)/t \qquad (2)$$

$$L_i = min_{\{0<t\leq T_i\}} L_i(t) \qquad (3)$$

$$L = max_{\{0<i\leq n\}} L_i \qquad (4)$$

式 (1) において、$L \leq 1$を満たす時、すべての処理のリアルタイム処理を保証できる。

(2) 拡張した Rate-Monotonic アルゴリズム

計算機 k において実行する処理 (1 ～ i) の計算量は式 (5) で表される。この式では、計算機間の計算能力の差を処理能力 (P_k) で正規化している。計算能力の指標としては処理速度 (MIPS 値) などを用いる。

$$W_{(i,k)}(t) = \frac{1}{P_k} \sum_{j=1}^{i} C_j \lceil t/T_j \rceil \qquad (5)$$

k:計算機番号　P_k:計算機kの計算能力

処理 (1 ～ i) の単位時間あたりの計算量は、以下の式で表される。

$$L_{(i,k)}(t) = W_{(i,k)}(t)/t \qquad (6)$$

以下に示す計算式で計算結果が 1 より小さければ、1 から i で示される処理まではリアルタイム処理を保証できる。

$$L_{(i,k)} = min_{\{0<t\leq T_i\}} L_{(i,k)}(t) \qquad (7)$$

以下に示す計算式で結果が 1 より小さければ、要求したすべての処理のリアルタイム処理を保証できる。

$$L_k = max_{\{0<i\leq n\}} L_{(i,k)} \qquad (8)$$

実現したグローバル・スケジューラでは、式 (8) を分散環境中の計算機 (k) ごとに適用し、リアルタイム処理を保証できる計算機を決定する。

3.2.3 スケジュール処理プログラムの構成

スケジュール処理プログラムは以下の 2 つの機能からなる。

(1) リアルタイムスケジュール機能
分散環境にある計算機の中から、リアルタイム処理を保証できる計算機を決定する。計算機の決定には、3.2.2節で述べた改良した Rate-Monotonic アルゴリズムを用いる。

(2) 処理分散機能
各計算機で走行する処理実行プログラムに処理の実行依頼を行う。

3.3 処理実行プログラム

処理実行プログラムは、スケジュール処理プログラムに依頼された処理をローカル・スケジューラに依頼する。この方式の場合、スケジュール処理プログラムと処理実行プログラムを非同期に実行すると、お互いの待ちが発生せず効率が良い。そこで、各計算機で動作する処理実行プログラムとスケジュール処理プログラムの間はメッセージ通信を用いて処理の実行依頼を行う。

4 まとめ

本報告では、Rate-Monotonic アルゴリズムを用いた分散リアルタイムスケジューラの実現方式について述べた。改良した Rate-Monotonic アルゴリズムを用いて、分散環境でのリアルタイム処理の保証を行うことが比較的容易に行え、このアルゴリズムを分散環境で使用できる可能性があることが分かった。

参考文献

[1] 神余浩大、藤山昌也、竹垣盛一, "協調分散スケジューリング方式のリアルタイム性保証", 信学技法、CPSY91 84(1991)

[2] John Lehoczky, 他, "The Rate Monotonic Scheduling Algorithm:Exact Characterization And Average Case Behavior" Technical Report Department of Statics Carnegie Mellon University 1987.