早稲田大学大学院理工学研究科

博 士 論 文 概 要

論 文 題 目

マルチプロセッサシステム上での
ダイナミックスケジューリング方式
に関する研究

申 請 者
合 田 憲 人
Kento Aida

電気工学専攻
情報システム制御研究

1996 年 12 月

近年，コンピュータの計算速度を向上させるためのアプローチとして，複数のプロセッサを接続してマルチプロセッサシステムを構成し，並列処理を行うことにより，プログラムの実行時間の短縮を図る方式が注目されている．特にマルチプロセッサシステムのハードウエア技術の向上はめざましい．しかしコンパイラやオペレーティングシステムなどのシステムソフトウエアにおける並列処理技術の遅れにより，ユーザがプログラムを実行した時の実質的な性能である実効性能が，ハードウエアのピーク性能に比べて著しく低いという問題がある．このような問題を解決する手段として，マルチプロセッサシステム上でタスクまたはジョブをプロセッサに割り当てる際のスケジューリング手法の開発が重要である．

マルチプロセッサシステム上でのスケジューリング手法は，タスクまたはジョブを実行する前に，全てのタスクまたはジョブのスケジューリングを行うスタティックスケジューリングと実行時にスケジューリングを行うダイナミックスケジューリングに分けることができる．スタティックスケジューリングでは，スケジューリングを行うタスクまたはジョブの情報が実行前に全て決定されている場合に，オーバーヘッドの少ない効率よい並列処理が可能である．しかし，単一プログラムの並列処理では，条件分岐等によりタスクの実行がプログラム実行時まで確定しない場合が多い．また，複数ジョブのスケジューリングでは，一般的にジョブがマルチプロセッサシステムに到着するまでジョブの情報を得ることができない．従ってこのような場合には，効率よいダイナミックスケジューリング手法が必要とされる．

しかし，従来のマルチプロセッサシステム上でのダイナミックスケジューリングを用いた並列処理では，次のような問題があった．単一プログラムの並列処理手法として用いられているループ並列処理やマルチタスキングでは，タスク間の並列性を十分に抽出できない，タスクのダイナミックスケジューリングオーバーヘッドが大きいという理由により，単一プログラムの効率よい並列処理が行えない．並列ジョブのスケジューリング方式として，動的にマルチプロセッサシステムに到着する並列ジョブに，到着順でジョブが要求する数のプロセッサを割り当てる手法が用いられているが，この手法では，プロセッサフラグメンテーションが大きく，プロセッサ利用率が低いため，効率よい並列ジョブのスケジューリングが行えない．

以上の背景を踏まえて，本研究では，単一プログラムの効率よい粗粒度並列処理手法であるマクロデータフロー処理を実用マルチプロセッサシステム上で実現し，マクロデータフロー処理手法が従来手法よりも有効性，実用性の高いことを実証した．次に，マルチプロセッサシステム上での並列ジョブのスケジューリング手法として，プロセッサフラグメンテーションを減少させ，ジョブを効率よくプロセッサへ割り当てる手法を提案した．また実用マルチプロセッサシステム上で提案手法の性能評価を行い，提案手法が従来手法よりも有効性，実用性の高いことを確認した．以下，各章毎にその概要を述べる．

1章「序論」では，本研究の背景を概観し，本研究の目的と位置付けを述べる．

2章「単一プログラムの並列処理」では，共有メモリ型マルチプロセッサシステム上で従来から実用または研究されている単一プログラムの並列処理手法について概説する．マルチプロセッサシステム上での単一プログラムの並列処理では，従来よりループ並列処理やマルチタスキングなどが用いられているが，これらの手法では，プログラムの並列性を十分に抽出できない，OSやランタイムライブラリによるタスクのダイナミックスケジューリングオーバーヘッドが大きいなどの理由から効率よい並列処理が行えないという問題があった．本章では，これら従来手法について述べるとともに，従来手法の持つ問題点を解決するために提案されている粗粒度タスク間の自動並列処理手法について述べる．

3章「マクロデータフロー処理手法」では，はじめに，単一プログラムの粗粒度並列処理手法として提案されているマクロデータフロー処理手法について述べる．本手法では，コンパイラが，プログラムの粗粒度タスク（マクロタスク）への分割，マクロタスク間の並列性抽出，マクロタスクをプロセッサへ割り当てるダイナミックスケジューリングコード生成，を自動的に行うことにより，マクロタスク間の並列性を最大限に抽出し，またダイナミックスケジューリングオーバーヘッドが小さいという特徴を持つ．

本章では，次に，マクロタスクのダイナミックスケジューリング方式として，スケジューリング処理を全プロセッサに分散させる分散スケジューラ方式と単一プロセッサに集中させる集中スケジューラ方式について述べる．分散スケジューラ方式では，コンパイラが生成するスケジューリングコードがユーザプログラムの一部として各マクロタスクの前後に挿入され，各プロセッサがマクロタスクを実行する度にスケジューリング処理を行なう．集中スケジューラ方式では，スケジューリングコードとユーザプログラムコードが別々に生成され，単一プロセッサがスケジューリング処理を行ない，他のプロセッサがマクロタスクを実行する．マルチプロセッサシステム上でマクロデータフロー処理を実現する場合には，使用するマルチプロセッサシステムのアーキテクチャ等を考慮し，これらのダイナミックスケジューリング方式のうちどちらかを選択して用いる．

本章では，最後に，マクロデータフロー処理の実用マルチプロセッサシステム上での性能評価について述べる．集中共有メモリ型マルチプロセッサシステムAlliant　FX/4および分散共有キャッシュメモリ型マルチプロセッサシステムKendall Square Research KSR1上でマクロデータフロー処理を実現し，性能評価を行った結果，マクロデータフロー処理がプログラムの実行時間を従来手法を適用した場合の1/1.92から1/8.10に短縮する等，本手法が従来手法に比べて有効性，実用性の高いことが確かめられた．

4章「マルチプロセッサシステム上でのジョブスケジューリング」では，マル

チプロセッサシステム上で従来から実用または研究されているジョブスケジューリング手法について概説する．マルチプロセッサシステム上での並列ジョブのスケジューリングでは，従来から逐次型計算機上で用いられているジョブスケジューリング手法は，タスク間同期によるビジーウエイト，キャッシュヒット率低下などの要因により，ジョブの効率よい並列処理が行えないという問題がある．また，並列ジョブに対してジョブが要求する数のプロセッサを割り当てる新たな手法も必要になる．本章では，はじめに，逐次型計算機上で用いられている手法をマルチプロセッサシステム上で用いる場合の問題点について述べるとともに，これら問題点を解決するために提案されているジョブスケジューリング手法について述べる．次に，並列ジョブに対してジョブが要求する数のプロセッサを割り当てる手法として提案されているジョブスケジューリング手法について述べる．

5章「並列ジョブのスケジューリング手法」では，マルチプロセッサシステムに動的に到着する並列ジョブのダイナミックスケジューリング手法であるFit Processors First Served (FPFS)およびFit Processors Most Processors First Served (FPMPFS)を提案する．本章で対象とするジョブスケジューリングでは，ジョブスケジューラが，各ジョブに対してジョブが要求する数のプロセッサを実行時に割り当てる．このようなジョブスケジューリングでは，各ジョブにプロセッサを割り当てる際に発生するプロセッサフラグメンテーションを小さく抑えることが，効率よい並列処理を実現するために重要である．提案手法であるFPFSでは，スケジューリング時にジョブキューを検索してその時点でのアイドルプロセッサ数に適合するジョブをプロセッサに割り当てることにより，プロセッサフラグメンテーションを小さく抑え，プロセッサ利用率を向上させる．またFPMPFSでは，ジョブキューの検索を行う前に，ジョブキュー内のジョブをジョブが要求するプロセッサ数をもとにソーティングすることにより，さらなるプロセッサフラグメンテーションの減少を図る．

本章では，次に，提案手法の性能解析とシミュレーションによる性能評価について述べる．性能解析では，待ち行列モデルと一次元Bin-packing問題のアルゴリズムを用い，提案手法が従来手法よりもプロセッサ利用率およびシステムの平衡条件を向上させることを示す．シミュレーションによる性能評価では，性能解析結果と同様に，提案手法が従来手法よりもプロセッサ利用率およびシステムの平衡条件を向上させることが確認され，提案手法の有効性が確かめられた．

本章では，最後に，提案手法の実用マルチプロセッサシステム上での実現方法と性能評価について述べる．マルチプロセッサシステムNEC　Cenju-3上で動作するジョブスケジューラを開発し，本ジョブスケジューラを用いて提案手法と従来手法の性能比較を行なった結果，提案手法が，従来手法に比べてプロセッサ利用率を9%～19%向上させるなど，有効性，実用性が高いことが確認された．

6章「結論」では，本研究で得られた成果と今後の課題について述べる．