Vol. 36 No. 8　　情　報　処　理　　Aug. 1995

解　説

日本におけるオペレーティングシステム研究の動向

2.4 メッセージプール指向の並列OS K1†

福　田　晃††

1. はじめに

マルチプロセッサ上の1つのユーザプログラムは，並列化コンパイラ（または実行時ルーチン）とOS[1]の助けを借りて並列実行される．従来の並列処理研究は，ユーザプログラム内の並列性をいかに抽出するかという並列化コンパイラの研究に重点がおかれ，並列実行される各部分（アクティビティ）の各々が発するシステムコールの実行は相変わらず逐次的であった．マルチプロセッサ用のOS（並列OS）が管理しなければならない計算機資源の数を増やさないようにすることは重要である．しかし，システムが大規模化するほど，この計算機資源は少なからず増加し，システムコールの処理には計算機資源の管理がともなうことが多いので，1つのシステムコールの処理が重たくなる可能性が高い．そこで，並列実行によりシステムコールの高速処理が必要と考える．

並列OS K1は，上記の背景にたって，ユーザプログラム内の並列実行はもちろん，OS内部の並列処理の可能性を追求するものである．

2. K1の設計方針と概要

2.1 設計方針

（1） 2つのレベルの並列処理の追求

1)ユーザプログラム内の並列処理と，2)OS内部の並列処理の2つのレベルの並列処理を追求する．特徴は，OS内部の並列処理の追求である．一般的に並列化には，入力となるデータ量に依存して並列度が決まり，それらをマージするなどして並列実行するデータ並列と，実行途中により動的に並列度が決まり，それらを並列実行するコントロール並列がある．従来のOSの多くは，ユーザプログラム内のアクティビティレベルでのみの並列実行であり，OSレベルでの並列実行はあまり考慮されていなかった．すなわち，ユーザプログラム内のアクティビティは，OS内部においても1つの制御フローとして実行されていた．K1は，ユーザプログラム内の並列処理と，アクティビティが発する1つのシステムコール内部の並列処理を含むOS内部の並列処理を追求する．

（2） 軽い並列実行環境

ユーザプログラム内のアクティビティの高速実行を図るため，ユーザ空間とカーネル空間で軽い並列実行環境を提供する．したがって，ユーザレベルおよびカーネルレベルで軽量プロセス（スレッド）を提供する．

2.2 K1の概要

K1は，マイクロカーネルアーテクチャを採用する（**図-1**）．マイクロカーネルは，メッセージプール機構，仮想プロセッサ管理，低レベルメモリ管理，割込みハンドラなどから構成される．K1マイクロカーネル上には，OSをエミュレートするシステムサーバ群がある．さらに，ユーザレベルスレッド管理機構により応用プログラムは実行される．ここでは，メッセージプール機構について述べる．

メッセージプールは，OS内部の並列処理のために設けられた機構であり，カーネル空間内の共有メモリ上におかれる．カーネル内およびシステムサーバ内のアクティビティは，メッセージプールを介して実行される．複数スレッドの並列生成など1つのメッセージを複数のアクティビティで処理することによりデータ並列，さらには，1つのアクティビティが異なる複数のポストにメッセージを送ることによりコントロール並列が実現できる．さらには，メッセージの型として，メッセ

† The K 1 Massage-Pool-Based Multiprocessor Operating System by Akira FUKUDA (Nara Institute of Science and Technology).

†† 奈良先端科学技術大学院大学情報科学研究科

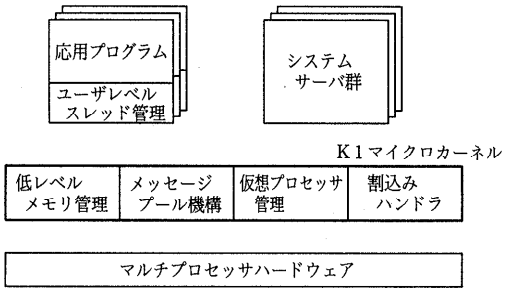


図-1　K1のソフトウェアアーキテクチャ

ージを送信したときに，メッセージ内容の処理が終るまで送信アクティビティがブロックされるブロッキング型と，メッセージ送信後，送信アクティビティも実行できる非ブロッキング型がある．非ブロッキング型メッセージによりさらなる並列処理が実現できる．

上記は，並列処理の枠組みを与えるものであり，これらを用いてOS内部の並列処理の可能性を追求する．

3. 研究課題とソフトウェア技術

3.1 研究課題

K1では，以下の研究課題がある．

（1） メッセージプール

メッセージ操作のオーバヘッドおよびメッセージプールへの競合の軽減方法が重要となる．また，メッセージの型として，非ブロッキング型の有効性およびその他の型についても研究する．

（2） システムサーバの構成

K1に限らず，マイクロカーネルアーキテクチャをとるOSでは，OS機能の大部分はシステムサーバでエミュレートされることになるので，システムサーバが重要な要素となる．並列性，拡張性，適応性などの観点からシステムサーバの分割方法，構造，並列実行モデルなどのアーキテクチャを研究する．

（3） ユーザ空間とカーネル空間との並列協調処理

ユーザプログラムの並列実行は，並列化コンパイラ（または実行時ルーチン）とOSによって行われるので，OSは常に並列化コンパイラを意識する必要がある．高度化する並列化コンパイラとOSの並列協調処理の可能性を追求する．

3.2 ソフトウェア技術

種々の課題があるが，現状のソフトウェア技術を述べる．

（1） データ並列処理

データ並列処理の典型に複数スレッドの並列生成がある．スレッドを複数生成する場合，メッセージプールに1つのメッセージを送ることによりスレッドを並列生成することになるが，その効率は，スレッドを生成するアクティビティのメッセージプールへの競合およびメッセージ処理のオーバヘッドとスレッド並列生成とのトレードオフとなる．シミュレーションによる種々の並列生成方式を評価した結果，メッセージプールへの競合およびメッセージ処理のオーバヘッドを軽減しキャッシュを活用する目的で，1つのメッセージアクセスで複数のスレッドを生成する方式を採用している[2]．

（2） アイドルプロセッサの管理方法

大規模マルチプロセッサでは，動的にしろアイドルプロセッサが生じる可能性が高いので，アイドルプロセッサの管理方法が重要となる．アイドルプロセッサが常にメッセージプールを検索する方法では，メッセージプールへの競合が生じ，システムの性能低下を引き起こす可能性が高い．アイドルプロセッサの管理方法として各種方式を評価した結果，アイドルプロセッサをキューとして管理する方式を採用している[3]．

4. 研究プロジェクトと開発現状

K1研究プロジェクトでは，大きく分けて以下の3つのサブプロジェクトがある．これらについて現状を述べる．

4.1 マイクロカーネルアーキテクチャ

（1） K1マイクロカーネル[4]

裸のマルチプロセッサWS BE上に，メッセージプールを含むK1マイクロカーネルを実装している．実装は，既存のマイクロカーネルをまずBEに実装した後，修正するという方法をとった．メッセージ型については，現在は，ブロッキング型メッセージのみしかサポートしておらず，予想されたことではあるが，メッセージプールにメッセージが溜りやすいという現象が生じている．メッセージプールの構造およびメッセージ処理等を含めて，現在改良中である．

(2) システムサーバ[5)~7)]

システムサーバの構成方式を研究している．現状は，その第一段階として，既存のシステムサーバを例にとって，単一サーバ構成，分割サーバ構成およびそれらの各種並行実行方式を実装し，評価している．

4.2 スケジューリング[8)~10)]

カーネル空間とユーザ空間との協調処理の第一段階として，両空間内のスケジューラによる2レベルスケジューリングを研究している．本スケジューリングの発想自体は，特に目新しいものではないが，本スケジューリングにおいて不十分であった系統的な方策整理およびその性能評価を，各種共有メモリアーキテクチャを対象として，シミュレーションを用いて行っている．また，さらに，分散共有メモリアーキテクチャでは重要となるメモリ管理との協調スケジューリング（Memory affinity scheduling と呼ぶ）の研究に着手している[10)]．

4.3 並列実行環境[11)]

ユーザ空間でスレッド管理を行うユーザレベルスレッドライブラリ PPL (Parallel Pthread Library) を作成している．スレッドライブラリとしてはすでにいくつかあるが，PPLは，異なる仮想プロセッサモデルへの容易な実装（移植性）とユーザレベルスレッドの並行実行ではなく並列実行（並列性）の2つを特徴とする．その第1版がいくつかのシングルプロセッサWSで稼働中であり，現在マルチプロセッサシステムに実装中である．

5. おわりに

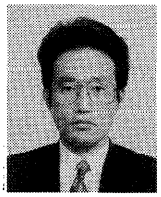
K1は，各レベルの並列処理を追求するものである．システムが大規模になればなるほど，少なからずOSが管理しなければならない計算機資源は増加するので，OS内部の並列処理の可能性を探ることは重要であると考える．K1プロジェクト内のサブプロジェクトは現在独立して進行しているが，今後これらを統合していきたい．

謝辞 BE WSの寄贈など，日頃サポートしていただいている松下電器産業(株)映像音響情報研究所画像情報グループチームリーダ 浅原重夫博士に感謝いたします．

参考文献

1) 福田 晃：並列オペレーティング・システム，情報処理，Vol. 34, No. 9, pp. 1139-1149 (1993).
2) Tsunedomi, K., Fukuda, A., Murakami, K. and Tomita, S.: A Message-Pool-Based Parallel Operating System for the Kyushu University Reconfigurable Parallel Processor—Parallel Creation of Multiple Threads—, J. of Information Processing, Vol. 14, No. 4, pp. 423-432 (1991).
3) 今村，桑山，宮崎，林，福田，富田：並列オペレーティング・システムK1の設計と実現，情報処理学会「並列処理シンポジウム JSPP'92」，pp. 305-312 (1992).
4) 桑山，最所，福田：並列オペレーティング・システムK1—マイクロカーネルの考察と設計，情報処理学会「コンピュータシステム・シンポジウム」，pp. 69-76 (1992).
5) 桑山，最所，福田：マイクロカーネル構成OSにおけるシステムサーバの分割数と性能，情報処理学会「コンピュータシステム・シンポジウム」，pp. 155-162 (1993).
6) Kuwayama, M., Saisho, K. and Fukuda, A.: A Scheme for Organization of System Servers in Microkernel-based Operating Systems and Its Performance, Proc. Joint Conf. on Software Engineering (JCSE'93), pp. 193-199 (1993).
7) Kuwayama, M., Saisho, K. and Fukuda, A.: Organization Scheme of System Servers in Microkernel-based Operating Systems—Multi-process and Multi-thread Methods—, Proc. COMPSAC'94, pp. 307-312 (1994).
8) 甲斐，藤木，福田：マルチプログラミング環境のマルチプロセッサにおける2レベル・スケジューリング—スケジューリング構造と性能評価—，情報処理学会論文誌，Vol. 35, No. 10, pp. 2115-2127 (1994).
9) Fukuda, A., Fujiki, R. and Kai, H.: Two-level Processor Scheduling for Multiprogrammed NUMA Multiprocessors, Proc. COMPSAC '93, pp. 343-351 (1993).
10) 甲斐，藤木，福田：メモリ管理と協調動作する2レベルスケジューリング，情報処理学会「並列処理シンポジウム JSPP'94」，pp. 319-326 (1994).
11) Miyazaki, T., Kuwayama, M., Saisho, K. and Fukuda, A.: Parallel Pthread Library (PPL): User-level Thread Library with Parallelism and Portability, Proc. COMPSAC '94, pp. 301-306 (1994).

(平成6年7月20日受付)



福田 晃（正会員）

1954年生．1977年九州大学工学部情報工学科卒業．1979年同大学院修士課程修了．同年NTT研究所入所．1983年九州大学大学院総合理工学研究科助手．1989年同大学助教授．1994年より奈良先端科学技術大学院大学情報科学研究科教授．工学博士．オペレーティング・システム，並列化コンパイラ，計算機アーキテクチャ，並列/分散処理，性能評価などの研究に従事．本会平成2年度研究賞，平成5年度Best Author賞受賞．訳書「オペレーティングシステムの概念」（共訳，培風館）．ACM，IEEE Computer Society，電子情報通信学会，日本OR学会各会員．