分散ＯＳ ＣＯＳＭＯＳ２における負荷分散

6P-1

李在己　計宇生　相田仁　斉藤忠夫
東京大学 工学部

１．はじめに

ＣＯＳＭＯＳ２は当研究室で実装されている統合型分散ＯＳである。これは従来ＯＳが持つ主な機能のプロセス管理、ファイル管理、入出力管理などに通信処理機能を加えて、これらの諸機能を論理機能モジュールとして分割し、高速なＬＡＮ上に結ばれている各ノードに分散配置する。このように分散された各論理機能モジュールは、ＣＯＳＭＯＳカーネルバックプレーン（ＣＫＢ）を介して相互協同することにより、システム全体を１つのＴＳＳのように動作させることができる。これにより、分散システムの特徴である、高価な資源の共有、機能と負荷の分散、高性能、拡張性、信頼性などが得られる。

本稿では、ＣＯＳＭＯＳ２における負荷分散の方法について述べる。

２．ＣＯＳＭＯＳ２の構成

図１は一般化したＣＯＳＭＯＳ２のシステム構成を示している。

ＣＯＳＭＯＳ２では、図１のように独立した論理機能モジュールを物理的な各ノードに適当に分散配置する。これによりネットワークトランザクションに柔軟に対応するノード構成となり、高度な負荷分散が可能である。

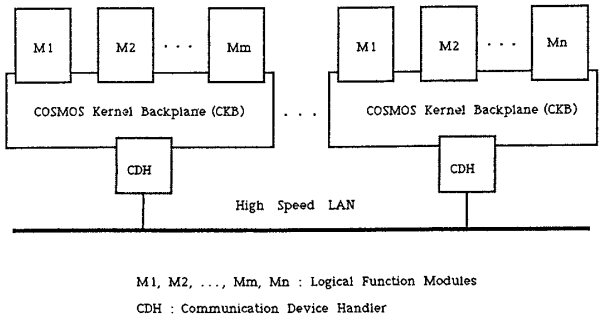

図１．ＣＯＳＭＯＳ２のシステム構成

各ノードを構成する論理機能モジュールはマネージャ、サーバとデバイスハンドラの３つに大別される。

マネージャモジュール（Process Manager, File Managerなど）はシステム内に１つのみであり、ネットワークワイドな各種資源と情報を統合管理する。　各種のサーバ（Process Server, File Server, Print Serverなど）はシステム内に複数存在することができ、ユーザからの要求に対して、各々指定されたサービスを提供するモジュールである。

デバイスハンドラ（Disk Device Handler, Communication Device Handlerなど）は各種の入出力機器を直接に制御する機能を持つモジュールである。[1]

ＣＫＢではノードの初期化、ノード内の論理機能モジュール間の通信とスケージュリング、及びＬＡＮを介したノード間の通信を司る。特に、ＣＯＳＭＯＳ２では、この２つの通信がユーザに透明に行われる。ＣＫＢに対して、各論理機能モジュールは１つのソフトウェアボードのような構成を成す。特に、ＣＯＳＭＯＳ２では、この２つの通信がユーザに透明に行われる。

通信デバイスハンドラは[2]ノード間のメッセージ通信に使われる部分である。ここだけを変更することにより、各種のＬＡＮに対応することができる。

３．ＣＯＳＭＯＳ２の負荷分散

負荷分散とは、システム中の各ノードが均等な負荷を持つようにすることである。これを用いることにより、システム内のアイドルノード或は軽負荷のノードと過負荷のノードに均等な負荷を割り当てて、システム全体の性能を上げることができる。

ＣＯＳＭＯＳ２では、ある意味で２段階の負荷分散を行っている。第一段階はシステム構築の際に、システムマネージャが静的にノードの構成を決めることにより行われる。これは機能分散による方法である。第２段階の負荷分散はプロセスサーバ内において、プロセスの実行時、動的に行われる方法である。[3]

ここではＣＯＳＭＯＳ２の第２段階負荷分散の方法と、これを実現するための関連論理機能モジュール間の動作について述べる。

Load Balancing in Distributed Operating System COSMOS2
Jaekee LEE, Yusheng JI, Hitoshi AIDA, Tadao SAITO
University of Tokyo

3．1 負荷の測定

ＣＯＳＭＯＳ２において、負荷分散に直接に携わる論理機能モジュールは、システム内の唯一なプロセスマネージャ（ＰＭ）と全プロセスサーバ（ＰＳ）である。

ＰＳはＵｎｉｘと同じ概念を持ち、プロセスの生成、実行、終了及びスケジューリングを行う。ＰＭにはマネージャプロセスがあり、このプロセスが各ＰＳとのトランザクションを行う。マネージャプロセスはＰＳからの要求を順次に処理する。

各ＰＳは自分の負荷と、システム内の全ＰＳが持つ負荷の平均値（これをシステム負荷と呼ぶ）を維持する。これに対し、ＰＭはシステム内の全ＰＳの負荷と、システム負荷を管理する。

次に単一ＰＳの負荷（η_p）とシステム負荷（η_s）の計算式を示す。

$$\eta_p = N_p / T_{ps} \quad \cdots\cdots\cdots \quad (1)$$

$$\eta_s = \Sigma \eta_p / P_n \quad \cdots\cdots\cdots \quad (2)$$

ここで、N_pはＰＳ上のユーザプロセスの実行待ち行列の長さであり、T_{ps}は１つのノード内において、全論理機能モジュールに対するＰＳの実行時間の割合である。そしてP_nはシステム中のＰＳの数である。

3．2 負荷分散の方法

ＣＯＳＭＯＳ２での負荷分散は、ＰＳ上のプロセスの実行時、式（１）と（２）により求められた自分の負荷とシステム負荷を比較して、ＰＭと相手のＰＳとのネゴシエーションにより決められる。これはリモート実行による負荷分散であり、リモート実行の必要条件は $\eta_p > \eta_s$ である。

図２はＣＯＳＭＯＳ２でのＰＳとＰＭ間のリモート実行の手順を示している。

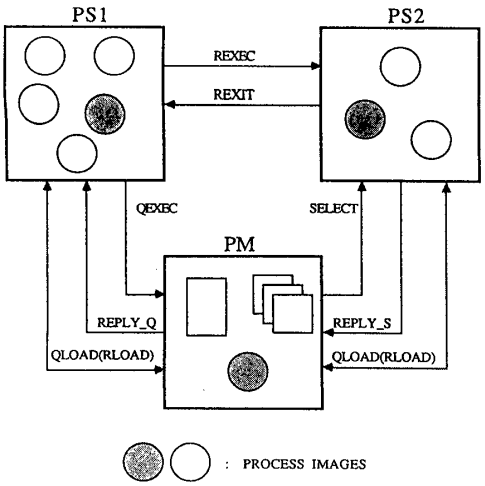

図２．ＣＯＳＭＯＳ２のリモート実行の手順

ここでＰＳ１からリモート実行の必要が生じた場合、ＰＳ１はＰＭに自分の負荷とともにリモート実行の要求メッセージを送る。これに対してＰＭは負荷管理テーブルを参照して、ＰＳの中で一番負荷が軽いＰＳを選ぶ。ＰＭは選択されたＰＳ（ここではＰＳ２）に現在のシステム負荷を送り、リモート実行の可否を要求する。ＰＳ２はリモート実行の可否を決めて、ＰＭに応答する。ＰＳ２から可の応答が来ると、ＰＭはリモート実行を要求したＰＳ１にリモート実行の宛先を送って応答する。　その後、ＰＳ１は直接にリモート実行をＰＳ２に依頼することができる。ＰＳ２での実行が終わると、実行の結果をＰＳ１に送りリモート実行を終了する。もしＰＳ２の応答が否なら、ＰＭは他のＰＳを選び、同じ処理を行う。

一方、ＰＭと各ＰＳ間の負荷の更新のやり方は、各ＰＳが一定時間（数秒）ごとに自分の負荷を計算して、ＰＭに報告する。ＰＭはこれに対する応答として、その時点のシステム負荷をＰＳに送る。一定時間内に、あるＰＳからの負荷の報告がなければ、ＰＭはそのＰＳに対してポーリングを行い、システムの負荷を維持する。

4．おわりに

以上のように、ＣＯＳＭＯＳ２での負荷分散はシステム内の全ＰＳの負荷をＰＭが効率的に管理することにより、リモート実行の際に動的に行われる。もしＰＭがダウンしても、各ＰＳによるローカルな実行は正常的に行われる。

この方法を拡張すると、システムのフォルトトレランスも容易に実現できる。

今後はこの方式の実装を進めて、なおノード間のモジュールの動的なマイグレーションについても検討するとともに、ＣＯＳＭＯＳ２の性能評価を行う予定である。

＜参考文献＞

［１］計、李、鈴木、相田、斉藤、”分散ＯＳ ＣＯＳＭＯＳ２の実装（１） －ＯＳ機能モジュール”、情報処理学会第３６回全国大会、４Ｄ－１、１９８８．３

［２］鈴木、李、計、相田、斉藤、”分散ＯＳ ＣＯＳＭＯＳ２の実装（２） －通信サブシステム”、情報処理学会第３６回全国大会、４Ｄ－２、１９８８．３

［３］松下、計、相田、斉藤、猪瀬、”分散オペレーティングシステムＣＯＳＭＯＳ２ －プロセス管理方式”、情報処理学会第３４回全国大会、４Ｂ－８、１９８７．３