5J-1

PC クラスタ向け OS「SSS-PC」のための負荷分散スケジューラの設計

久松 佳之† 松本 尚‡ 並木 美太郎§
東京農工大学 工学部 情報コミュニケーション工学科 †
国立情報学研究所 ‡
東京農工大学 大学院 共生科学技術研究部 §

1 はじめに

近年、パーソナルコンピュータ（PC）のコスト低下を背景に PC クラスタリング技術の研究が進んでいる。その代表的なものに OpenMosix[1] や Score[2]、そして SSS-PC[3] が挙げられる。中でも SSS-PC は、単体マシンから 10 万台規模の PC やサーバーまでを束ねて使用可能な次世代オペレーティングシステムとして注目されている。

SSS-PC はその上で動作するプログラムによる自律的な負荷分散 [4] を特徴の一つとしている。このため、複数のノード間で情報を同期させたりタスクを移送させる仕組みは整っているが、システム全体として負荷を分散させる仕組みは無く、プログラムの自主性に任されている。しかしプログラム毎に負荷分散を考えていたのでは無駄が多い。そこで、本研究では SSS-PC のノード毎に働く負荷分散スケジューラの設計を行った。

2 目標

本研究は SSS-PC において負荷分散のためのスケジューラを設計することを目標とする。

このスケジューラーは他ノードと比較して高負荷なノードにおいて、移送可能なタスクを他のノードにマイグレートさせることにより、ノード間の負荷バランスを平均化することを目的とする。そのために、SSS-PC の情報開示機構などを活用してノードの負荷情報やタスク情報の収集を行う。また、それらの情報をもとにマイグレーションの必要性の判断や移送させるタスクの選定を行い、マイグレーションが必要であると判断した場合は該当タスクにマイグレーションを行わせるためのシグナル発行を行う。

これにより、タスクの特定ノードへの偏りを防ぎ、システム全体として性能の向上への寄与が期待できる。

3 全体構成

SSS-PC 全体においてどのように負荷分散を行うかを図 1 に示す。

ノード間同士では絶えず情報開示機構によるノード情報の交換を行っており、本スケジューラを含むノード内のタスクは情報開示機構より全ノードの負荷情報を低コストで得ることができる。本スケジューラは、その情報を元に負荷の小さいノードへいくつかのタスクをマイグレーションさせる。

本研究によって開発するスケジューラシステムは以下のような構成となる。

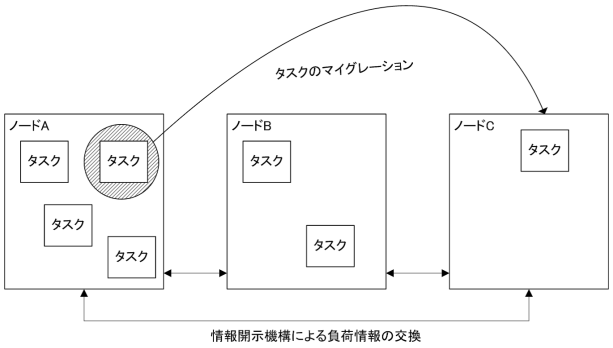

図 1: SSS-PC 全体における負荷分散の仕組み

- スケジューラ（ノード毎に 1 つ）
- シグナルをキャッチする機構（各プログラムに組み込む）

ノード内でのスケジューラとタスクの関係を図 2 に示す。

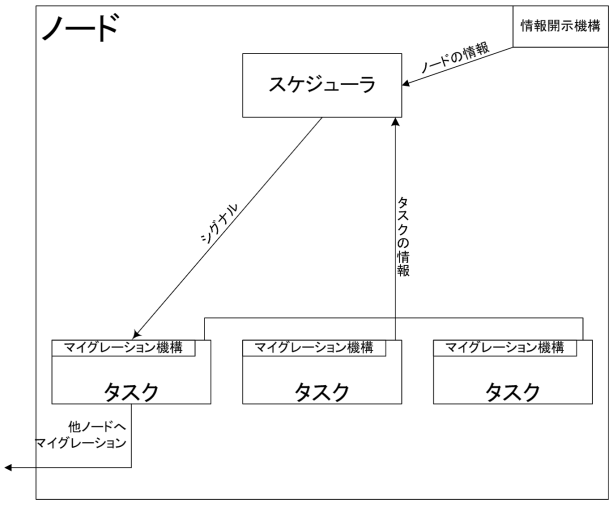

図 2: ノード内におけるスケジューラの働き

スケジューラは通常のタスクと同様に起動してノード毎に常駐し、負荷情報の収集、判断、及びタスクへのシグナル発行を行う。プログラム側でシグナルをキャッチし、タスクのマイグレーションを実行するための機構はライブラリとして組み込む。これは、SSS-PC 上ではマイグレーションは全て自発的に行われなければならず、スケジューラが強制的にタスクをマイグレーションさせることは不可能であるからである。

Design of a load balancer on the OS "SSS-PC" for a PC cluster

† Yohisyuki Hisamatsu
Department of Computer, Information and Communication Sciences, Tokyo University of Agriculture and Technology
‡ Takashi Matsumoto
National Institute of Informatics
§ Mitaro Namiki
Department of Computer, Information and Communication Sciences, Tokyo University of Agriculture and Technology

4　スケジューラ

スケジューラは以下のような機能を持つ。

1. ノード毎・タスク毎の情報の収集
2. マイグレーション実施の判断・対象タスクの選定
3. マイグレーション対象タスクへのシグナル発行

ノード毎の情報は SSS-PC に備わっている情報開示機構から取得する。情報開示機構から取得できる情報のうちノードの負荷に関わるものを表 1 に示す。

種類	備考
ノードに存在するタスク数	ready/wait/sleep
CPU クロック数	MHz 単位
CPU 負荷	short/midlle/long
メモリ容量	全体・空き容量

表 1: 情報開示機構で取得できる情報（一部）

ノードに存在するタスク数は状態によって ready・wait・sleep の 3 種類に分別される。また、CPU 負荷は一定時間内に CPU に存在する ready 状態のタスク数を表し、時間の長さによって short（1 秒）・middle（10 秒）・long（100 秒）の 3 種類の情報が取得できる。メモリの容量は物理メモリ全体の容量と空きメモリの容量が取得できる。

タスク毎の情報は SSS-PC に備わっているシステムコールによって取得する。

5　マイグレーションの可否の判断及び実行

これまでに得た情報により、スケジューラは自ノードから他ノードへマイグレーションが必要かどうか、及びどのタスクをマイグレーションさせるかを決定する。マイグレーションの可否は、以下の手順によって判断される。

5.1　負荷の小さいノードの選択

スケジューラは、情報開示機構から一定時間後とに情報を取得し、各ノードの CPU 負荷を比較する。

CPU 負荷の 3 種類の値をそれぞれ CPU クロック数で割り、「1MHz 単位の CPU 負荷」を 3 種類算出する。そして、それぞれの値が最も低いノードを選び出す。選び出された 3 つのノードのうち 2 つ以上が一致した場合、そのノードを「CPU 負荷の小さいノード」とする。

2 回連続で同一のノードが CPU 負荷の小さいノードであると判断された場合、次のステップへ移行する。

5.2　マイグレーションの可否の判断

5.1 で選択された最も CPU 負荷の小さいノードと自ノードの CPU 負荷を比較し、自ノードの方が負荷が大きい場合はマイグレーションを行う決定を行い、タスクの選択に移る。タスクの選択については次節で述べる。

自ノードの方が負荷が小さい場合はマイグレーションを行わないので 5.1 に戻る。

5.3　マイグレーション対象となるタスクの選択

スケジューラが「マイグレーションが必要」と判断した場合、マイグレーション対象となるタスクの選定に移る。

まず、マイグレーションが不可能なタスクを対象から除外する。これにはスケジューラ自身、また SSS-PC のシステムに関係するタスク（nikomon shell など）が該当する。

残ったタスクのうち以下の数値を基準に対象のタスクを決定する。

(1) タスクの priority 値

タスクの priority 値（優先度）が低いタスクほど、他ノードへのマイグレーションのために一時的に中断しても問題がないタスクと判断して優先的にマイグレーション対象とする。

(2) タスクの life 値

タスクの life 値（どれだけの時間生存しているか）が短いタスクからマイグレーション対象とする。

(1)(2) の基準を元にマイグレーション対象タスクに順位を付け、順位が一番高いタスクをマイグレーション対象のタスクとする。

5.4　マイグレーションコストとの兼ね合い

マイグレーション先ノード、マイグレーション対象タスクが決まったら、そのタスクをマイグレーション先ノードに送って良いのかどうかの判断を行う。

SSS-PC 上においてマイグレーションを実行すると、ネットワークに GigabitEthernet を用いたクラスタシステムにおいて 60msec 以上、100BASE-TX を用いたシステムではさらにその倍近い 110msec 以上の時間がノード間の移送にかかる。マイグレーションを行うかどうかの決定はこの時間的コストを考慮に入れ、自ノードの負荷がそれほど大きくなく移送する価値が無い場合はマイグレーションを行わない。

また、メモリ上におけるタスクのサイズが相手先ノードの空きメモリ容量を圧迫しないかどうかも検討する。タスク自体のサイズが大きい場合はマイグレーションのコストも大きくなり、十ミリ秒単位で違ってくることもある。

情報開示機構からのデータ取得には大きな時間的コストはかからないため特に考慮する必要はない。

6　タスク側の動作

スケジューラはマイグレーション対象タスクにシグナルを送ってマイグレーションを実行させる。

タスクは、スケジューラからのシグナルをキャッチした場合にマイグレーションを行う。SSS-PC ではマイグレーションをシステムコールを呼び出すことによって行える。入出力は最初に起動したノードに残るため、ユーザー側がマイグレーションを意識することなくプログラムの実行を継続することができる。

7　おわりに

本研究では、SSS-PC のための負荷分散スケジューラの設計を行った。これにより、SSS-PC を用いた並列処理プログラムの負荷分散に寄与し、結果として性能の向上に期待することができる。

参考文献

[1] OpenMosix : http://openmosix.sourceforge.net/
[2] Score : http://www.pccluster.org/
[3] SSS-PC : http://www.ssspc.org/
[4] 松本尚, 平木敬 : 自由市場原理に基づくスケジューリング方式 , 電子情報通信学会技術研究報告 CPSY-99 (SWoPP'99), Vol. 99, No. 251, pp. 63-70, August 1999