システムソフトウェアと オペレーティング・システム 74−25
マルチメディア通信と分散処理 81−25
（1997. 2. 28）

非均質マルチプロセッサシステムにおける Nレベルスケジューリング方式

松澤伸一，藤井章博，根元義章
東北大学大学院 情報科学研究科
〒980-77 宮城県 仙台市 青葉区 荒巻字青葉
E-mail:shin@nemoto.ecei.tohoku.ac.jp

あらまし　多数の計算機を通信ネットワークを用いて結合し構成される仮想並列マシンは，大規模な演算能力を提供しうるため，膨大な計算を必要とする科学技術計算の分野で有効なツールである．仮想並列マシンではその効率を向上させるため，構成している個々の計算機の能力が一様でない，動的にシステムの処理能力が変化する，などの環境を考慮したスケジューリングが行われる必要がある．しかし，これまでのスケジューリングに関する研究では，各計算機が有する計算機資源について考慮されておらず，その性能も均一であるという前提で議論がなされている．そこで本稿では，より現実に則した仮想並列マシンのモデル化を行うために，多数の計算機資源を考慮し，様々な種類の計算機，およびタスクが存在する非均質システムのモデルを導入する．ついで，その上で効果的な並列タスクの動的なスケジューリング機構を提案する．本方式の有効性をシミュレーションを用いて評価した結果，タスクの平均実行時間が減少する効果が確認できた．

N-level Scheduling Scheme for Heterogeneous Multicomputer System

Shin-ichi MATSUZAWA, Akihiro FUJII, and Yoshiaki NEMOTO
Graduate School of Information Sciences, Tohoku University
Aoba-ku, Sendai-shi, Miyagi Pref., 980-77 JAPAN
E-mail:shin@nemoto.ecei.tohoku.ac.jp

Abstract　Parallel Virtual Machine consist of multicomputers is one of the emerging alternatives for large scale computations. In such systems, variable types of machines with different architecture are used and dynamic changes of status occurs. Effective scheduling of parallel and sequential tasks is important issue in this environment.

In this report, we introduce models of heterogeneous computing environment that represents real multicomputer systems and propose a dynamic scheduling mechanism over the model. The simulation results shows that the average response time of parallel processing is reduced by using the proposed method.

1 はじめに

多数の計算機を通信ネットワークで結合し構成される仮想並列マシンは，計算機台数を増やすことにより，その処理能力を高めることができる．また科学技術計算のように膨大な計算量を必要とする場合，タスクを複数の計算機に分散して並列処理することで処理時間の短縮化が実現できる．従って，この様な分野において仮想並列マシンは有効なツールとして期待されている．

仮想並列マシンは疎結合型のマルチコンピュータシステムであるが，これは(1)個々の計算機の処理能力は一様ではない，(2)システム処理能力の動的な変動が激しい，という状況が一般的な利用環境であり，並列タスクをどの計算機に割り当て処理させるかを決定するスケジューリングが重要である．

(1)の状況はいわば，システムが非均質であることを意味する．仮想並列マシンにおいて各計算機の処理能力の差は比較的大きく，入出力装置やグラフィックエンジンなど個々のノードがもつ計算資源も多様である．この非均質性を考慮し，最適なコストパフォーマンスを得るためには，システムの性質を十分考慮したスケジューリングを行う必要がある．従来のスケジューリングに関する研究では，主に超並列システムを対象にしているため，システム内の計算機の処理能力は均一であると仮定しているものが大半であり [1,3]，また非均質を仮定していても，CPU 能力の差のみに限定されている [2,6]．仮想並列マシンにおいては，各々が独立の計算機であるため比較的処理能力の差が大きいと考えざるを得ない．よって，従来のプロセッサの処理能力は一定と考えるモデル化では負荷が平等にならず効果が少ないという問題が指摘できる [5]．

(2)の状況は，システム状態の動的な変動がスケジューリング結果に及ぼす影響を考察する必要がある．静的スケジューリングだけでは有効な結果が得られないためシステムの変動に追従してタスクの割り当てを行う動的なスケジューリングが必要である [5,6]．

以上のように，多様な計算資源を要求するタスク群に対して，適切な計算機群を動的に効率よく配分するスケジューリング機構についての検討は十分ではない．そこで，本稿では効率の良いスケジューリング方式の確立を目指す．

本稿の構成は，まず2.では，ネットワーク，I/O，グラフィック能力などの多様な計算資源を考慮した非均質システムを取り扱うシステムモデルを示す．3.では，そのモデル上で適切なスケジューリングを実現する方式を提案する．4.では，提案したスケジューリング方式の有効性をシミュレーションを用いて評価し，5.でまとめる．

2 モデルの定義

ここでは，タスクとシステムの不均質性として計算資源を考慮したモデルの定義を行う．

2.1 システムのモデル

本研究では，並列タスクの実行環境を，①処理スピードの差，② 通信遅延，③動的なシステム能力の変動，さらには④個々の計算機がもつ資源群，の4つの要素を考慮しモデル化する．

計算機の資源としては，CPU 能力，通信能力，I/O 能力，グラフィック性能，記憶装置の能力など様々なものが考えられるが，いまネットワークシステム全体でN種類存在すると仮定する．各々の計算資源はその有無のみを考慮し、性能の絶対値は考慮しないものとする．

計算機が保有する資源は式 (1) の N 次元ベクトルで表され、ベクトル内の変数は計算機に対してある資源 (γ_{ij}) の有無を表す．即ち変数は，1または0のみを取るものとし，これが1のときは資源がその計算機上で利用できることを表し，逆に0のときは利用できないことを表す．例えばグラフィック WS では、他の WS と比べてグラフィック性能が高いと判断され，それに対応する資源の値が1となる．

$$\Gamma_i = \{\gamma_{i1}, \gamma_{i2}, ..., \gamma_{iN}\} \quad (1)$$

$$\gamma_{ij} = \begin{cases} 1 & (\text{資源 j が利用可能}) \\ 0 & (\text{資源 j が利用不可能}) \\ * & (\text{未定義}) \end{cases} \quad (2)$$

システム中に存在する計算機台数を M 台として，仮想並列計算機 ($\mathcal{VM}$) 中の計算機 P_i を式 (3) の4項組で表す．

$$\mathcal{VM} = \{(P_i, W_i(t), T_i, \Gamma_i)\} \ (i = 1, ..., M) \quad (3)$$

ここで変数$W_i(t)$,T_iは以下の通りである.

- $W_i(t)$ $[I/s]$:計算機P_iのCPU能力
  個々の計算機の1秒間に処理できる命令(I)数. 動的な負荷の影響により時間変化する.
- T_i $[bps]$:計算機P_iの通信能力:
  個々の計算機のI/Oも含むた通信能力.
- Γ_i:計算機P_iの保有資源ベクトル
- M [台]:総計算機台数

2.2 並列タスクのモデル

一方タスクは, 処理するために必要な命令数と通信データ量の他に, そのタスクを処理するために必要な資源の種類をコンパイラが検出した要求資源ベクトルΔ_iにより表す.

$$\Delta_i = \{\delta_{i1}, \delta_{i2}, ..., \delta_{iN}\} \quad (4)$$

$$\delta_{ij} = \begin{cases} 1 & (\text{タスクiは資源jを必要}) \\ 0 & (\text{タスクiは資源jを不必要}) \end{cases} \quad (5)$$

要求資源ベクトルΔ_iは, 資源δ_{ij}を要素とする. δ_{ij}はその資源をタスクが必要とするときは1, 必要でないときは0の値をとるものとする. タスクは, それが要求する資源の組に合致した資源をもつ計算機のみで実行可能であると仮定する. ここで並列タスク($\mathcal{T}$)を, 以下の5項組で定義する.

$$\mathcal{T} = \{(t_i, I_i, D_i, L_i, \Delta_i)\} \quad (i = 1, 2, ...) \quad (6)$$

ここで,

- t_i:タスク
- I_i $[I]$:命令数
  タスクt_iを解くために必要な全命令数.
- D_i $[bit]$:データ量
  全転送データ量. 初期転送データ, 結果転送, 計算途中でのメッセージなど全て含めた量.
- L_i [個]:タスク分割数
  タスクを分割する量. 並列度.
- Δ_i:要求資源ベクトル

3 スケジューリング方式

ここでは, 2.で示したタスクとシステムのモデルを用いて, タスク側で要求する資源ベクトルに合致する資源ベクトルをもった計算機群を動的にかつ適切に割り当てる新しいスケジューリング方式を提案する.

3.1 2レベルスケジューリング方式

提案方式の前提となる2レベルスケジューリング方式(以下2LSC方式)について示す. 2LSC方式は, 比較的大規模なマルチプロセッサシステムに対して有効であることが甲斐ら[3]により報告されている. この構造を図1に示す.

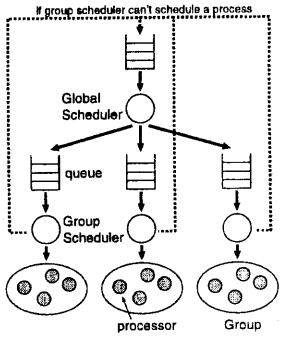

図1: 2レベルスケジューリング方式の構造

スケジューラは各々タスクを格納するキューをもつ. また, グループ内に属する計算機を管理しタスクの割り当てを行う下位スケジューラ(グループスケジューラ)と, 下位スケジューラを管理しグループ間でタスクの移送を行う上位スケジューラ(グローバルスケジューラ)に分類される. 以下に各スケジューラの動作を示す.

(1) グループスケジューラの動作

グループスケジューラはタスクが到着すると, まずそのタスクが要求する資源ベクトルと合致する計算機群を自分が管理する計算機グループ内から選択する. また, グループに属する計算機の状態とキューに貯められたタスク数から, グループ全体の負荷状態を測定する. 測定した負荷が高いと判断された場合や合致する計算機数がタスクの分割数に満たない場合はタスクをグローバルスケジューラに移送する. そうでないならば自分のグループ内でタスクを処理する.

(2) グローバルスケジューラの動作

グローバルスケジューラは, グループスケジューラから移送されてきたタスクを他のグループに転送する役目を担う. 具体的には各グループの

負荷を定期的に測定し，それらの中で一番負荷の低いグループを選択してタスクを移送する．

なお，本稿で採用する2LSC方式では，計算機群はCPU能力の差によりグループ分けされているものとする．

3.2 資源管理二分木とスケジューリングマネージャの導入

前節で示した2LSC方式では，システムの複雑化やシステムの高負荷化に伴い，グローバルスケジューラの負担が急激に増加することが予想される．これは，タスクの実行時間を増加させる大きな原因である．よって本稿では，スケジューラにかかる負担を軽減するNレベルスケジューリング方式を提案する．

どの計算機に対しても，各資源の有無が各々独立であるならば，全ての計算機群を保持している資源で2^N個のグループに分類できる．この様に分類された計算機グループの中から，あるタスクを実行するために適切なグループを探索することで，タスクの総実行時間の短縮が可能である．本文で提案する方式は不均質システムにおける資源管理を二分木に基づいて行う．

ここで，多数の計算機グループを二分木を用いて階層的に分類する．この木は図2，図3に示すように，N個の資源によって$(N+1)$段で構成され，木のi段目のノードが，資源γ_iを保有しているか否かに従って$(i+1)$段目で2つの子ノードに分類される．

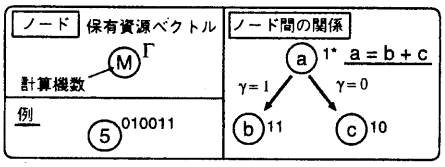

図2: 資源管理二分木の定義

図2において，ノード内の番号が計算機の台数を，右肩の数字列が保有資源ベクトルを表す．

スケジューリングマネージャ($\mathcal{M}$)は，二分木内の各ノードに対して一つづつ配置される．各マネージャは，ある資源ベクトルを保有する計算機グループを管理する．以下，二分木のレベルiに属するマネージャをレベルiマネージャと呼ぶ．レベルiマネージャ($M_{i,k}$)は，まず自分の保

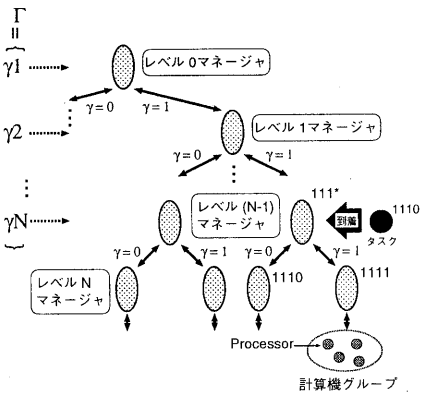

図3: Nレベルスケジューリング方式の構造

有資源ベクトル(Γ_k)，サブグループ(子供以下のグループ)に属する全計算機台数(m_k)，2つの子マネージャ(レベル$i+1$マネージャ)のアドレス($ac1_k, ac2_k$)，さらに親マネージャ(レベル$i-1$マネージャ)のアドレス(ap_k)を保持している．

$$\mathcal{M} = \{(M_{i,k}, \Gamma_k, m_k, ac1_k, ac2_k, ap_k)\} \quad (7)$$
$$(i = 1, ..., N, k = 1, ..., 2^N)$$

3.3 Nレベルスケジューリング方式の提案

前述の資源管理二分木とスケジューリングマネージャを用いた，分散管理型のNレベルスケジューリングアルゴリズム(以下NLSC方式)を以下に示す．このアルゴリズムは，各マネージャ上で，そのマネージャにタスクが到着した事をトリガとして起動される．

1. **(タスクの到着)**
   レベルK($K \leq N$)マネージャ$M_{K,i}$においてタスクt_jの到着が確認されたら2.へ進む
2. **(実行条件の判定)**
   マネージャは，その保有資源ベクトルΓ_iとタスクの要求資源ベクトルΔ_jを比較する．もし，$\gamma_{ik} = \delta_{jk}$ $(\forall k | 1 \leq k \leq K)$でかつ，$K = N$ならば5.へ，$K < N$ならば4.へ，それ以外ならば3.へ進む．
3. **(親マネージャへのタスクの移送)**
   $\gamma_{il} = \delta_{jl}$ $(\forall l | 1 \leq l \leq K-1)$でかつ，$\gamma_{il+1} \neq \delta_{jl+1}$である場合，タスクを親マネージャ$M_{K-1,h}(h < i)$に移送する．

4. (子マネージャへのタスクの移送)
タスクの要求資源ベクトルにより，2つの子マネージャの内から，δ_{jk+1}側の子マネージャを選択し，タスクを移送する．
5. (計算機へのタスク割り当て)
グループの中から負荷の低い計算機をL_j台選択し，それぞれにタスクを移送する．

4 性能評価

ここでは，3. で提案したNLSC方式の有効性をシミュレーションにより評価する．

4.1 性能指標

本稿では，動的なスケジューリング方式の評価指標として，タスクの平均実行時間を用いる．これは，タスクが発生してから実行終了するまでの時間で，計算機上でタスクが実際に処理される時間に加え，タスクが計算機に割り当てられるまでの時間 (スケジュール待ち時間) も含まれる．

4.2 シミュレーション条件

シミュレーション条件は以下の通りである．

- **システム構成:**
N種類の資源間の依存関係はない．すべての資源はM台の計算機に対してそれぞれ等しい確率で存在する．
- **タスクの発生:**
タスクについても同様に，各資源に対して全て等しい確率で要求する．またタスクは，それが要求する資源ベクトルと関係なくある計算機グループにランダムに発生する．
- **タスクの移送:**
タスクは，それが発生したグループ内で処理できない場合には，適切なグループに移送されなければならない．

[パラメータの設定]

- 総計算機台数M:96 [台]
- 資源数N:4 [種類]
- タスクの分割数L:4 [個]
- シミュレーション時間:10^6 [u.t.]
- システム負荷:5〜70 [%]

4.3 シミュレーション結果

シミュレーション結果を図4に，システム負荷が低い部分の拡大図を図5に示す．ここで図中2LSC方式において，Gは計算機グループ数を表す．

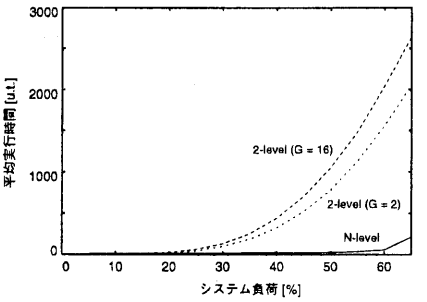

図 4: 平均実行時間の比較結果 (1)

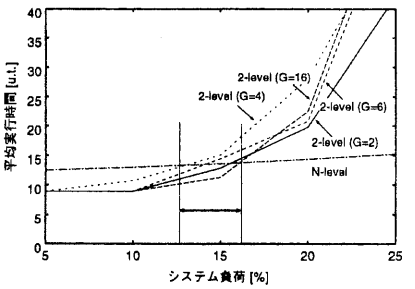

図 5: 平均実行時間の比較結果 (2)

システム負荷が低い (5〜10%程度) とき，2LSC方式の方が約1.3倍優れた結果となっている．この原因は，NLSC方式では，そのタスクが発生したグループ内で処理されることがほとんど無く，タスクの移送にオーバーヘッドを要するためと考えられる．

しかしこの傾向は，システム負荷を15%前後に変化する．2LSC方式ではシステム負荷の増加にしたがって急激に平均実行時間が増加する，即ちスケジューラの性能は低下する．一方，NLSC方式ではほとんど性能の低下は見られない．即ちこの範囲で二つのスケジューリング方式の性能の逆転が生じる．

さらにシステム負荷が高くなると，この性能差はますます広がって行く．この原因は，2LSC方式では，グローバルスケジューラが既にボトルネッ

クになっており，ここでのスケジューリング待ち時間が平均実行時間の大半を占めるが，NLSC方式ではスケジューラの量が多いためにタスクが一つのスケジューラに集中せずボトルネックは生じていないためである．よって，NLSC方式の優位性が明らかになった．

4.4 静的スケジューリングも含めた評価

さらに提案するNLSC方式に併用して，静的スケジューリングを実施した際の効果に関する性能評価も行った．ここでいう静的スケジューリングの効果とは，予めタスクに必要な資源を予測し，それらを有するグループに実行前に割り当てることができるというものである．これにより，保有資源に着目してグループ分けを行うNLSC方式では，適切な計算機グループの近くにタスクを発生させることができ，結果としてタスク移送回数の減少が期待できる．

図6に4種類の計算機資源の内，2種類の計算機資源に関しては静的な解析が可能とした場合におけるシミュレーション結果を示す．

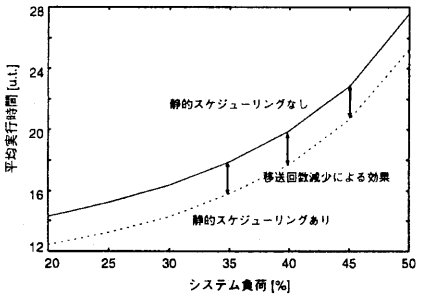

図6: 静的スケジューリングも含めた評価

この結果から，静的スケジューリングの効果が現れていることが分かる．平均実行時間の減少分は，タスクが移送されるマネージャ数が減少したために生じたものと言える．この差は，移送コストが大きくなるにつれて広がると考えられる．よってNLSC方式では，静的スケジューリングを効果的に組み合わせることにより，さらにスケジューリングオーバーヘッドを減少させることができる．

5 まとめ

本稿では，高速なLANなどのネットワーク環境で仮想並列マシンを構成することを前提とし，多様な計算機資源を取り扱うための方策を検討した．複雑な環境に対して適切な計算機群を動的にかつ効率良く選択するためには，スケジューリングに要するオーバーヘッドを分散させ，効率を高める必要がある．提案したNレベルスケジューリング方式は，スケジューリングオーバーヘッドを分散でき，結果としてタスクの平均実行時間を低く押えることができる．

さらに，静的スケジューリングを併用した場合を考慮したシミュレーションでは，Nレベルスケジューリング方式は，動的な変化の少ない計算機資源に対しては，予め適切な配分が可能であり，結果としてさらに実行時間の短縮が計れることが分かった．

以上の点から，複数の計算機資源を考慮した仮想並列マシンにおける動的なタスクのスケジューリング機構として，Nレベルスケジューリング方式の有効性が認められた．

今後の課題としては，本方式を実際の演算処理問題に適用するための検討などが必要である．

[参考文献]

[1] *A.M.Tilborg,L.D.Wittie, 'Wave Scheduling -Decentralized Scheduling of Task Forces in Multicomputers',IEEE Transaction of Computers, Vol.C-33,No.9,(1984)*

[2] *Chin LU,Sau-Ming LAU, 'An Adaptive Load Balancing Algorithm for Heterogeneous Distributed Systems with Multiple Task Classes', In Proceedings,The 16th ICDCS,pp.629-636,(1996)*

[3] 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・スケジューリング方式-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,(1994)

[4] 水野, 李, 石井, '非均質マルチプロセッサシステムにおける通信時間を考慮したスケジューリング手法', 情報処理学会研究会報告,96-HPC-15,pp.87-92,(1996)

[5] 松澤, 藤井, 根元, 'LAN環境における並列処理に関する研究', 情報処理学会研究会報告,96-HPC-25,pp.147-152,(1996)

[6] 松田, 藤井, 根元, '仮想並列マシンにおける負荷分散に関する研究', 情報処理学会 マルチメディア通信と分散処理ワークショップ, pp.121-128,(1996)