Vol. 31 No. 7 情報処理学会論文誌 July 1990

分散オペレーティングシステムにおけるプロセス移送の方式†

朴 圭成†† 芦原 評††
清水 謙多郎†††* 前川 守††

プロセス移送はネットワーク透過な分散システムを実現する上で重要な機能であり，性能面のみならず多くの面で有用である．本論文はプロセス移送の方針（policy）に重点を置き，解析解および詳細なシミュレーションを通じて様々な方式を横断的に比較・評価した．まず，情報の精度が性能に与える影響の解析では，情報の精度が極めて重要であることが明らかになった．プロセス移送の方針として数多くの提案が過去になされているが，解析の結果，移送判定の主体は受け手主導，移送判定の時期は負荷情報の変化に即応する方式，移送プロセスの選択は残留時間最大のものを選ぶ方式（したがって，プロセス中断を伴う），移送先は負荷最小のノードを選ぶ方式が一般に優れていることが明らかになった．さらには，残留処理時間による負荷の評価，プロセス移送回数に制限を設けないことが性能を向上させることが明らかになった．この方式は GALAXY 分散オペレーティングシステムにも採用される．

1. まえがき

プロセス移送とは分散システムにおいてプロセスをノード間で移送し，動的に再配置する機能をいう．プロセス移送の目的を整理すると次のようになる．

（1） プロセスの応答時間の短縮
（2） スループットの向上
（3） 最大スループットの向上
（4） ハードウェア・ソフトウェア資源の有効利用
（5） ファイル等のレプリカの数の削減
（6） 特定のジョブの応答時間の短縮
（7） 信頼性の向上
（8） 可用性の向上
（9） 保全性の向上

このように，プロセス移送の目的は多岐にわたり，ネットワーク透過な統合化された分散システムを実現する上でプロセス移送の機能はきわめて重要である．また，技術的には，ネットワーク通信速度の向上およびプロセッサ処理速度の向上により，移送に要するオーバヘッドが軽減され，プロセス移送の有効性はますます増大する傾向にある．

一般に，プロセス移送は，移送先のノードおよび移送するプロセスを決定する段階と，実際にプロセス移送を実行する段階の二つに分けられ，それぞれプロセス移送の方針（policy）および機構（mechanism）の問題として論じられる．従来のプロセス移送の研究の多くは，方針に関するものである．しかしながら，これらのほとんどは特定の条件のもとで理論解を求めることを目的としたもので，議論の対象が限定され，現実への適用も困難なものが多い．それ以外の研究は，実動するシステムの実現に関する研究であるが，個々の OS 上での機構が議論の中心となっている．

本論文では，プロセス移送の方針に重点を置き，様々なプロセス移送アルゴリズムを横断的に比較・評価するとともに，シミュレーションを用いて確認を行う．また，プロセス移送を効果的に行うための指針を提示し，評価する．本章では以下，プロセス移送の方針に関して従来の研究の概括を行い，次に，本論文の前提となるプロセス移送の機構を提示する．

1.1 プロセス移送の方針

プロセス移送の方針は負荷情報をどのように収集するか，そしてその情報を用いて具体的にどのように移送先や移送プロセスを決定するかによって分類される．**表1**は，プロセス移送の方針として考えられるものを項目ごとに整理したものである[1)~17),19)~22),24)]．

情報収集の方式は，局所方式と大域方式に大別することができる．局所方式は状態情報の交換を行わず，したがって，プロセス移送の決定はすべてそのノード自身のローカル情報によって行う．

大域方式はシステム全体の情報を収集し，それに基づきプロセス移送を決定，実行する．

大域アルゴリズムを簡略化する方式は数多く存在する．しかし，基本的には，情報交換の範囲を特定のノードに限定するパーティション方式，情報交換の頻

† Process Migration Policies in Distributed Operating Systems by KYU-SUNG PARK, HYO ASHIHARA, KENTARO SHIMIZU and MAMORU MAEKAWA (Department of Information Science, Faculty of Science, University of Tokyo).
†† 東京大学理学部情報科学科
* 現在 電気通信大学情報工学科

度を限定する方式，過去の情報をもとに予測を行う方式とに分類することができる．

具体的なプロセス移送の方針は次の4個の決定からなる．

1）誰 (who) が移送判定の主体となるか

2）いつ (when) 移送判定を行うか

3）どの (which) プロセスを移送するか

4）どこへ (where) 移送するか

情報収集の方針および具体的なプロセス移送の方針は相互に強く影響を与えるものと比較的独立なものとがある．与えられたシステムの条件に対し最適な組合せを見出すことが本論文の目的である．

1.2 プロセス移送の機構

プロセス移送では，(a)プロセスの各種状態の移送，および(b)アドレス空間の移送が必要である．

(a)は，レジスタの待避内容やスケジューリングに関する情報，記憶管理用テーブル，入出力状態（入出力要求列，入出力バッファの内容，割込み信号），プロセスがアクセスするオブジェクトの一覧表などから成る．これらは，プロセスの実行に最低限必要な情報であり，移送が決定した時点で一括して転送すべきものである．

(b)については，一般に大量のデータの移送を伴うため，一括して転送する方式ではプロセスの実行が長時間にわたって中断され，プロセス移送の効果が減少してしまう．このため，ページ単位にデータを転送する方式が考えられる[23),25)]．ページ転送の方式としては，アドレススペースを前もって転送するプリページングと後から送るオンデマンド・ページングの二つの方式が考えられる．プリページングは，無駄な転送を行う危険性をもつものの，ワーキング・セットを考慮して一括した転送を行うことにより性能向上が期待さ

表1　プロセス移送方式の比較
Table 1　Comparison of process migration policies.

		情報収集の主体	情報収集のタイミング	移送判定の主体	移送判定のタイミング	移送判定の基準	移送先の決定	移送プロセスの決定
局所方式		なし	なし	移送元	定期的	局所的な閾値（移送元は負荷が閾値よりも大きいか，移送元は負荷が閾値よりも小さいかを調べる．）	無作為	任意
					プロセスの到着時など		距離が最小のもの	平均余命最大のもの
				移送先	定期的			大きさの最小のもの
					プロセスの消滅時など			
大域方式	情報収集と移送判定は基本的に独立	全ノードが互いに情報交換	定期的	移送元	定期的	局所的な閾値（固定および可変）	無作為	任意
		プロセス到着ノードが通知	プロセスの到着時など		プロセスの到着時など		距離が最小のもの	平均余命の最大のもの
		プロセス消滅ノードが通知	プロセス消滅時など	移送先	定期的			
					プロセスの消滅時など			
	移送判定時に情報収集	全ノードが互いに情報交換	移送判定時	移送元	定期的	全ノードの負荷の平均と自ノードとの比較		
		プロセス到着ノードが問合せ			プロセスの到着時など		現在の負荷が最小のもの	大きさの最小のもの
		入札			入札			
		全ノードが互いに情報交換			定期的	ヒューリスティック		
		プロセス消滅ノードが問合せ		移送先	プロセスの消滅時など		ヒューリスティック	ヒューリスティック
	パーティション方式	部分ノード間	全ノード間の場合と同様の分類が成り立つ．部分ノード群は，固定と動的に決定（移送判定時に無作為に選択）の二つが考えられる．対象範囲を限定することによって，移送コストを最小にするノードおよびプロセスが決定できる．					

れる．オンデマンド・ページングは，転送後，必要なページが揃うまでに時間がかかるものの，無駄な転送を行わなくてもよいという利点がある．実際には，両者の中間の方式，すなわち参照される可能性は高いがその参照が比較的後になりそうなページを前もって転送する一方，実際にアクセス要求が出されたページを最優先で転送する方式が考えられる．この場合，最適の転送方式を定めるには別途解析が必要である．また，こうしたページ単位の転送を実現するには，ノード間にわたる参照時コピーの機構が要求される．主記憶に存在するページ，ページング・ディスクに存在するページ，まだアクセスされておらずしたがってファイルとして存在するものなど，場合に応じた独自の転送機構も必要である．これらについては，現在筆者らが検討を進めているが，本論文のようなレベルで方針を論じる限り，上のような転送方式の詳細に立ち入る必要はないと考えられるので，ここでは転送コストのみを考慮し，転送方式の議論は省略することとする．

プロセス移送に際しては，移送前と移送後でプロセスの実行環境をいかに維持するかが問題である．プロセスの実行環境としては，

（a） そのプロセスと通信しあう他のプロセスおよびカーネル

（b） プロセスがアクセスするファイルやデバイスなどのオブジェクト

がある．本論文では，両者は区別しないで扱うことにする．すなわち，(b)のような一般の入出力をサーバプロセスとの通信とみなし，すべてプロセス間通信として統一して扱う．オブジェクトの移送は，通信相手のサーバプロセスの切替えとして捉えられる．

実行環境の維持については，オブジェクトの存在位置だけでなく，オブジェクトにアクセスする側（プロセス）の存在位置にも依らない位置独立なオブジェクトのアクセス機構が必要である．システムコールや一般の入出力に対しても位置独立性が保証されなくてはならない．また，プロセス自身の位置の変化がオブジェクトのアクセス機構に反映されなければならない．これまでプロセス移送を実現したシステムで，こうした要求を完全に満足するものはほとんどない．我々が開発中の GALAXY では，大域的なオブジェクト識別子に基づく位置づけ機構により，これを実現している．GALAXY では，プロセスを含むあらゆるオブジェクトのアクセスは，オブジェクト識別子を指定して行われ，オブジェクト・アクセスの位置独立性が統一的に実現されている．システムコールは，すべてプロセス間通信機構を通して行われる（真のシステムコール，すなわちカーネル呼出しはメッセージの送信と受信のみ）ので，時刻やノードの状態検出に関するようなノード依存の呼出しを除いて，すべてネットワーク透過に実行される．

また，プロセス移送の重要な課題として，過渡状態に対する処置がある．すなわち，移送の全過程を通して実行が矛盾なく継続されることが必要である．過渡状態の問題としては，オブジェクト・アクセスをすべてプロセス間通信とみなした場合，次のような場合が生じる．

（a） 移送されるプロセスにメッセージが送信され，これが受信される前にプロセス移送が行われた．

（b） カーネル呼出しの処理中に待ち条件が生じ，そのためプロセスの切替えが起こり，その間にプロセス移送が行われた．

まず，(b)については，カーネル呼出し処理中はプロセス移送禁止とし，その条件が解除されるまでプロセス移送の実行を停止する方式を用いるのが最も自然と考えられる．GALAXY のように，カーネル呼出しがメッセージの送受信に限定されるシステムでは問題はない．

(a)では，プロセス移送後に到着したメッセージは転送 (forwarding) を行う必要がある．この転送を可能にするための方式として，プロセスの移送先を移送元に残した方式が考えられる．この場合，転送のための情報をいつ消去するかが問題となる．プロセス移送を最初に実現した DEMOS/MP[18] では，プロセスの実行が終了するまで消去を行わず，プロセスの位置づけは，常に生成ノードから移送の履歴を示すリンクを辿って行う方式を採用した．生成ノードの位置はオブジェクト識別子に記載される．この方式の問題点は，(1)リンクを維持する操作が余計にかかる，(2)移送が連続したとき，プロセスへのアクセス時間がリンクの長さに比例して増大する，(3)リンクの途中のノードがダウンすると，アクセスが困難になる，などの点にある．望ましいのは，過渡状態におけるあらゆる未処理の操作がなくなったことが確認された時点で転送のための情報を消去する方式であるが，これはメッセージの到着順序等も考慮しなくてはならないため，実現は容易ではない．リンクをプロセス終了時まで保持する方式を用いても，プロセスの位置づけ機構を別にもたせることによって，上記(2)および(3)の問題

点を解消することが可能である．GALAXY では先程述べたオブジェクト識別子に基づく位置づけ機構により，オブジェクトの存在位置をアクセスする側のノード上でローカルに知ることが可能である．

プロセスの実行環境の維持および過渡状態に対する措置は，本論文で直接扱うべき内容ではないが，例えば，GALAXY のような方式を実現の一例として，以下の議論における前提条件としたい．

2. モデルと解析

性能面から見たプロセス移送とは，一言で言えばシステム内のノードの負荷を均等化することにより応答時間を最小化することと言える．そもそも負荷の不均衡が生じる原因には2種類ある．一つは定常的に存在するもので，これを本論文では定常的不均衡と呼ぶこととする．もう一つは時間の推移に従って負荷が増減することによって生じるものであり，これを動的不均衡と定義することとする．この両者は共にノードの負荷の不均衡を生じさせるものであるが，異なった理由に起因する．

2.1 定常均衡モデル

定常均衡モデルは待行列網としてモデル化できるが，通常の待行列網よりかなり複雑であり，待行列網に対して知られている唯一の一般解とも言える Jackson の定理はたとえポアソン入力，指数サービス時間を仮定しても適用不可能である．しかし，通信時間を零と仮定しての性能改善の限界は極めて簡単に得られる．

待行列の解析でよく知られているようにジョブの平均待時間はノードの負荷の増大に対して急激に増加するため，全体の負荷が一定の場合，ノード間の負荷を均衡させることにより，ジョブの平均待時間を最小化することができる．したがって，性能の上限は各ノードが独立の待行列を有し，そして各ノードの負荷が均等化されたモデルの性能により与えられる．（ノードの処理能力が異なる場合は重みをつけた負荷の平均化を行う必要がある．）

2.2 動的均衡モデル

負荷が飽和に近いノードが存在する場合には定常的不均衡を是正することは大きな性能改善となる．しかし定常負荷が均衡している場合でも実際には各ノードの待行列の長さは常に時間的に変動しており，各瞬間においてはノード間の不均衡が存在する．この不均衡をも平均化することにより，さらなる性能の改善が期待される．こうした動的均衡の性能は，システム内の全ジョブが一本の待行列を作るモデルの性能に相当し，これにより性能の上限を知ることができる．

表2にポアソン入力，指数サービス時間を仮定した場合の性能上限の比較の例を示す．これから明らかなように定常的均衡により性能が改善され，動的均衡により，さらに性能が改善されることがわかる．定常的均衡が達成されたあとでも，動的均衡による性能向上はノード数が多い場合特に著しく，動的均衡の必要性を十分に裏づけている．

2.3 均衡の精度と性能向上

さて，現実システムでは，プロセス移送に要する遅れ，システムの状態を把握するための通信の遅れなどのために，完全な均衡を行うことはできない．したがって，均衡の精度と性能改善の関係を知る必要がある．このような動的な均衡化を行った場合の性能は一般には解かれていない．ここでは，システムが取りうるすべての状態を考えてその間の遷移確率を求め，マルコフ鎖を構成して数値計算により各状態の定常確率を求める解析的手法をとる．解析は付録に示した．プロセス移送の方針は状態間の遷移の形として表される．均衡の精度をノード内の最大ジョブ数と最小ジョブ数の差 d をパラメタとしてノード数2に対して性能改善を示したのが**図1**である．これから高い精度が要求されることが観察される．特に $d=2$ から $d=1$（完全均衡）において最も大きな改善が見られる．**表3**は $d=1, 2, 3, \infty$ に対して示したものである．ノード数が増加するほど均衡による効果が大きいことも観察される．

表2 定常均衡，動的均衡による応答時間の減少
Table 2 Decrease of the response time by the stationary load balancing and the dynamic load balancing.

システム負荷	ノード数	ノード負荷	プロセス移送なし	定常均衡	動的均衡
0.5	4	半数は 0.04 他は 0.56	2.06	2.0	1.09
	32	半数は 0.406 他は 0.594	2.15		1.00
0.9	4	半数は 0.84 他は 0.96	16.3	10.0	2.97
	32	半数は 0.806 他は 0.994	94.3		1.14

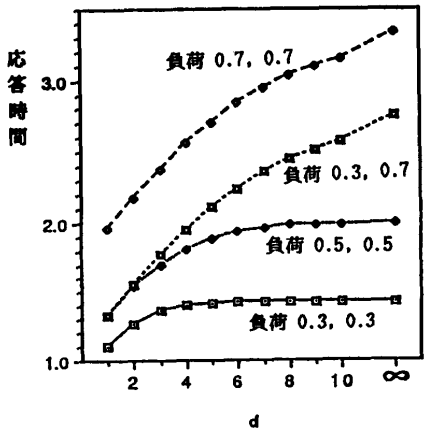


図 1 情報精度と応答時間（ノード数2）
Fig. 1 Relation between the response time and the accuracy of information (for two nodes).

2.4 第2章まとめ

プロセス移送をモデル化し解析的に解くことは極めて困難であるが，本節では性能改善の限界を見極めるために待行列網モデルとマルコフ鎖モデルを用いて解析を行った．この結果，次のような観察結果が得られた．

- 定常的均衡により性能向上が得られる．
- 動的均衡により，さらに向上する．
- 負荷に関する高精度の情報が必要である．
- ノード数が多いほど効果は大きい．

3. シミュレーション

3.1 モ デ ル

解析解では解析不可能な多くの事項について評価を行うために詳細な動的モデルを構築し，シミュレーションを行う．

仮定

- 各ノードでのプロセスのスケジュリングは多段階のラウンドロビン方式である．
- サービス時間として指数分布と標準偏差が平均の2倍であるような超指数（hyperexponential）分布の2種類を解析した．
- 通信回線の転送時間は指数分布を仮定した．
- プロセス移送による応答時間の変化，プロセス移送時間は現在の負荷をもとに待行列論に基づき予測する．

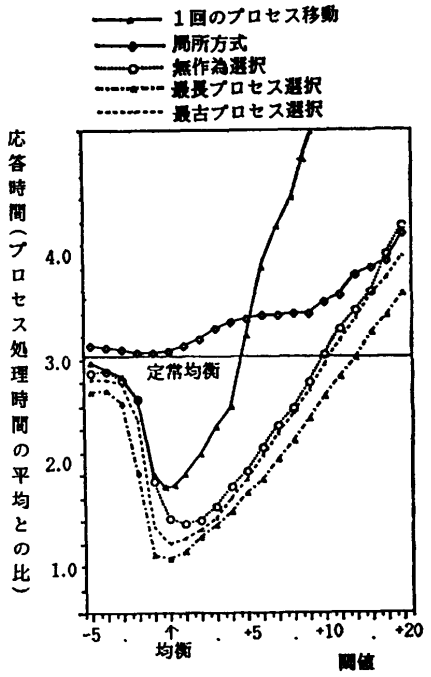


図 2 定常均衡，動的均衡による性能改善
（ノード数16，2グループの不均衡：負荷1.0, 0.0，正確な負荷情報，プロセス処理時間は超指数分布，プロセス移送時間の平均はプロセス平均処理時間の5分の1，送り手主導，即応型，移送先は最小負荷ノードの一つ）
Fig. 2 Performance improvement by the stationary load balancing and the dynamic load balancing.

システム状態の収集

システムの状態はメッセージを交換することにより収集する．

表 3 情報精度と応答時間（ノード数 2, 4, 8）
Table 3 Relation between the response time and the accuracy of information.

d		1			2			3			∞
ノード数		2	4	8	2	4	8	2	4	8	
システム負荷	0.1	1.01	1.00	1.00	1.09	1.10	1.10	1.11	1.11	1.11	1.1
	0.3	1.10	1.01	1.00	1.27	1.24	1.23	1.36	1.35	1.35	1.4
	0.5	1.33	1.09	1.02	1.54	1.38	1.34	1.70	1.60	1.57	2.0
	0.7	1.96	1.36	1.11	2.18	1.68	1.49	2.38	1.97	1.82	3.3
	0.9	5.26	2.97	1.88	5.48	3.29	2.62	5.70	3.62	2.65	10.0

プロセス移送の方式

局所方式，パーティション方式，大域方式について解析した．

3.2 定常的均衡，動的均衡の効果

均衡の効果を示したのが**図2**である．横軸はプロセス移送を起動する閾値であり，ノード内のジョブ数である．

- 動的均衡の中では大域方式が局所方式より優れている．また，プロセス移送の回数を制限しない方がよい．
- 閾値は性能に大きな影響を与える．
- プロセスの残留処理時間の長いものを移送した方がよい．

3.3 移送先の決定

図2において局所方式は無作為に移送先を定めている．これは大変性能が悪い．この結果および表2，3，図1は高精度の情報を集め，転送時間を考慮しながら負荷が最小のものに移送することが最善であることを示している．

3.4 メッセージの交換頻度（情報の精度）

メッセージ交換は頻繁なほど情報の精度は高くなる．メッセージ交換の頻度をパラメタとして比較したのが**図3**である．これから表3に示したのと同じく高精度な情報の必要性が明らかである．

3.5 パーティション方式の評価（情報の精度）

システムをいくつかのパーティションに分割し，プロセス移送をそれぞれのパーティションで行ったときの性能は局所方式と大域方式の中間が期待される．**図4**はパーティションの大きさによる影響を示したものである．これから大域的な情報を得るほど性能が高いことがわかる．しかもその差はかなり大きい．

メッセージ交換の頻度は時間的精度を示すものであるが，パーティション方式は空間的精度を表すと言える．これらは共に重要であることが示される．

3.6 移送判定の主体

送り手主導は過大移送，受け手主導は過小移送をもたらす傾向がある．不均衡が存在する場合両者の比較をしたのが**表4**である．受け手主導は移送回数を大きく減らすことが観察される．入札方式は，プロセス移送の開始は送り手主導，相手先決定は受け手主導（ま

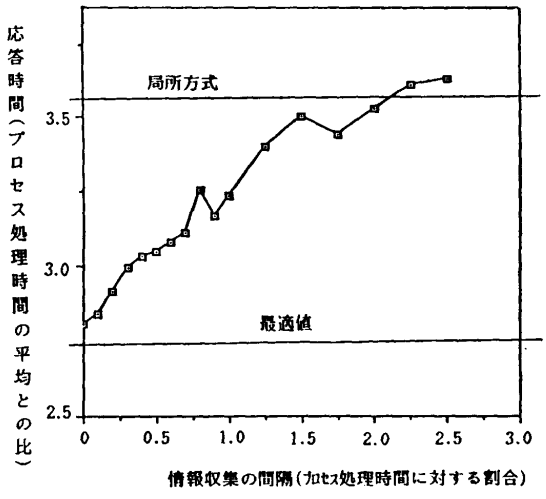


図3 情報収集の頻度
（情報収集の間隔は変化，最古プロセス選択，他は図2と同じ仮定）
Fig. 3 Relation between the response time and the frequency of information exchange.

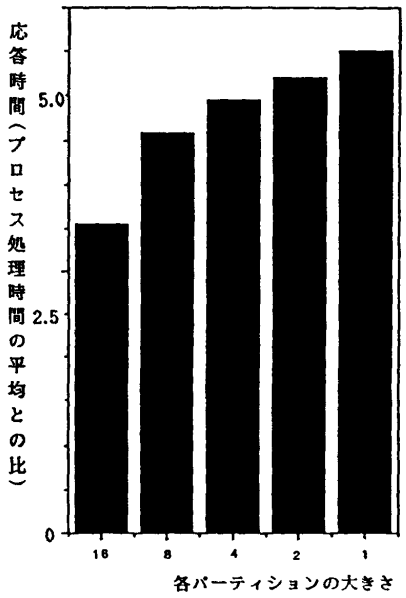


図4 パーティション方式
（他は図3と同じ仮定，ただし，パーティション内は最新情報）
Fig. 4 Relation between the response time and the number of partitions.

表4 受け手主導と送り手主導
（応答時間はプロセス処理時間の平均との比）
Table 4 Comparison of the response time and the number of migrations between the receiver-initiative method and the sender-initiative method.

方式	局所	送り手主導	受け手主導	送り手主導（相乗り）	受け手主導（相乗り）
応答時間	2.336	1.307	1.160	1.204	1.184
移送回数	4,147	1,642	81	868	59

たはその逆）と解釈することができる．言わば，送り手主導と受け手主導の欠点を相補う方式といえる．しかし，入札に伴うメッセージ交換のオーバヘッドは問題である．入札方式の性能は受け手主導とほとんど差はない．

3.7 プロセス移送の決定時期

プロセス移送の起動はプロセスの到着，プロセスの終了時に行うこともできるが，一定間隔で行うこともできる．その間隔を変化させたのが**図5**である．明らかに間隔を長くすると性能は低下する．

3.8 振動現象の解析と対策

解析解で解析が非常に困難なものに動的な振舞がある．特に振動現象がそれである．これを解析するために意図的に不均衡を導入し，その後の振舞を調べたのが**図6～図8**である．

- 情報収集に遅れがある場合は振動回数はかなり大きく，しかも長引く（図6）．もし，プロセス移送を1回に限定すると不均衡が長く続くこととなる．

次のような振動の防止策について解析した．

a）プロセス移送を実際に行う前に相手先の現在の負荷を再度確認して判断を下す．これは送り手主導の方式に，プロセス移送要求の返答に相手ノードの負荷情報を相乗りさせるという工夫をとり入れたものである（図7）．

b）軽負荷になったノードがプロセス移送の要求を送り，その要求を受け取ったノードのみプロセスを

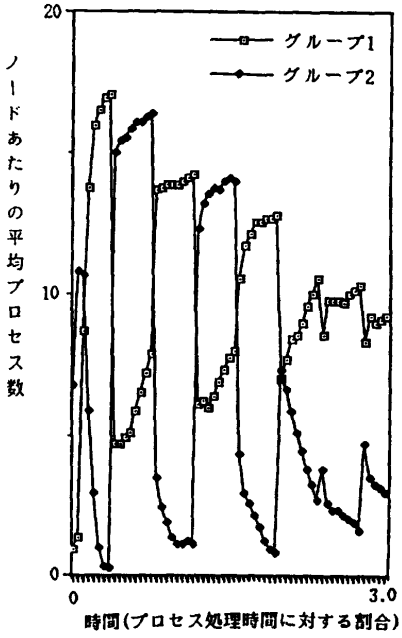


図6 振動現象
（メッセージ遅れはプロセス処理時間の5分の2，最古プロセス選択，他は図2と同じ仮定）
Fig. 6 Instability.

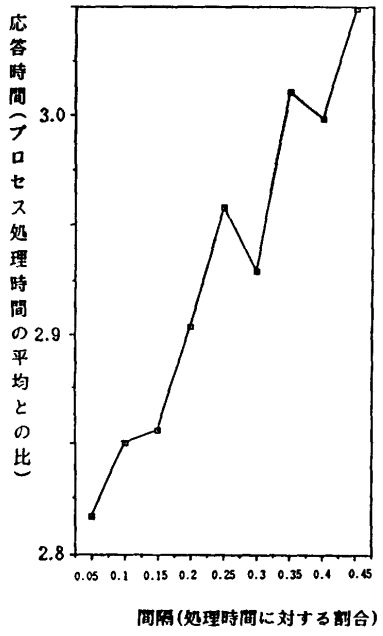


図5 プロセス移送の間隔
（最古プロセス選択，他は図2と同じ仮定）
Fig. 5 Relation between the response time and the frequency of migration decisions.

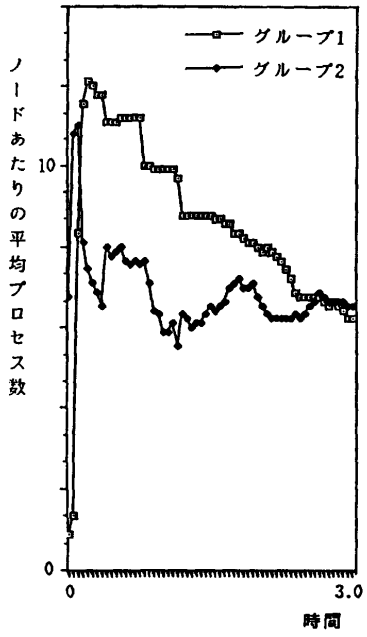


図7 送り手主導における振動現象
（メッセージ相乗り方式，他は図6と同じ仮定）
Fig. 7 Reduction of instability by the sender-initiative method.

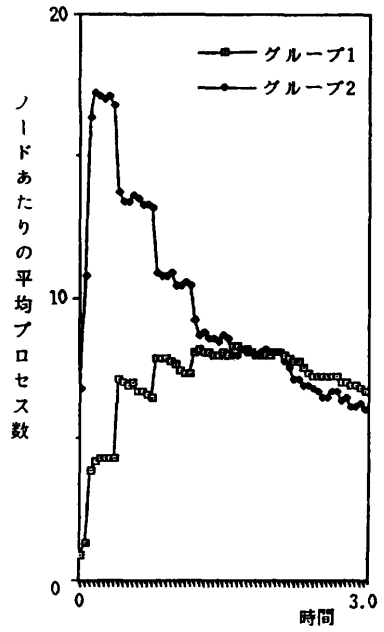


図 8 受け手主導による振動現象
(他は図6と同じ仮定)
Fig. 8 Reduction of instability by the receiver-initiative method.

移送する．受け手主導の方式である（図8）．
共に振動を減少させるのに役立つが，(b)の受け手主導方式の方が効果は大きい．

3.9 移送プロセスの選択

多段のラウンドロビンスケジュリングにおいて，移送するプロセスの選択を下位のキュー（処理時間の長いプロセス）から選択する方式と，無作為選択方式とを比較したのが**表5**である．

- 長時間プロセスを移送することにより応答時間が減少する．
- プロセス移送の数も減少する．

このことから長時間プロセスに対するプリエンプションを行うことが望ましいことがわかる．

3.10 考　　察

★性能向上／低下をもたらす要素を整理したのが**図9**である．左は情報の精度に依存するもの，右は依存しないものである．まず，図の左側における送り手主導は過大移送（振動を含む），受け手主導は過少移送をもたらす．情報の精度が高くなればこうした現象は緩和されるが，一般に受け手主導が振動も防止し，望ましい．入札は両者の欠点を補うものである．情報の精度が高くなれば，受け手主導は入札と同じ性能を得る．したがって，定期的に情報交換を行う受け手主導と入札方式は基本的には差はなく，得られた情報を蓄積する分だけ情報交換方式の方が優れていることとなる．しかし，プロセス移送がまれにしか生じない状況では情報交換は無駄となる．図9の右側は情報の精度に無関係に性能向上をもたらす項目を並べたものである．

表 5 ラウンドロビンキューの段数による性能向上
Table 5 Performance improvement for the number of round-robin scheduling queues.

段　　数	1	2	4	8
無作為選択方式に対する性能向上（%）	4.51	6.05	9.95	9.95

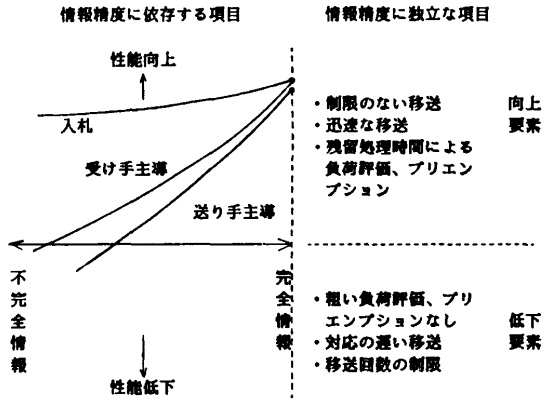


図 9 情報精度と性能向上要素との関係
Fig. 9 Relation between the accuracy of information and various factors of performance improvement.

- 状態変化に即応した移送．
- 制限のない移送．移送回数の制限は性能低下をもたらす．
- 残留時間による負荷の評価，プロセス選択，スケジュリング，プリエンプション．これは応答時間のみならずプロセス移送回数を減少させる．

これらの項目はお互い独立であり，その効果は相乗的である．

★**図10**は移送の頻度と移送1回当りに要する情報収集のオーバヘッドとの関係を示したものである．入札は毎回かなりの数のメッセージ交換を必要とする．したがって，定常的な情報交換方式の方がある頻度以上ではオーバヘッドは少なくなる．パーティションは部分集合であるためオーバヘッドは少ない．局所方式は情報収集のオーバヘッドはない．さて，情報の精度は時間的にも空間的にも高いことが望ましい．オーバヘッド削減の工夫としてはプロセス転送や負荷情報の相乗り，効率のよいルーティング等がある．

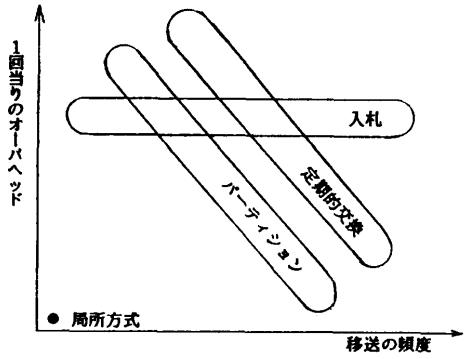


図 10 移送の頻度と情報収集オーバヘッドとの関係
Fig. 10 Relation between the frequency of migrations and the overhead of the information exchange.

4. プロセス移送の方針

システム負荷は短時間の間に大きく変化するのみならず，長時間の間にも，ゆっくりと変化する．これらに対処できるためにも動的均衡機構が必要となる．

具体的な施策はシステムの条件により異なる．しかし，オペレーティングシステムとしてはこれら条件の変化に対処できるように柔軟に構築されている必要がある．したがって，汎用機構として次の機構を提示する．この方式は実際 GALAXY 分散オペレーティングシステムで採用される方式である．

システムの分割

各ノードはそのノードからプロセスが移送される可能性のあるノードの集合を知っている．これをプロセス移送グループと呼ぶ．プロセス移送グループは互いに重なっていてよい．GALAXY ではグループ内での情報交換を分散方式で行うが，特定の制御ノードを仮定する方式も考えられる．本論文のこれまでの議論は特定のグループ内を対象とするものと考えられる．

表 6 応答時間改善度
Table 6 Improvemnt of the response time for each migration algorithm.

移送方式 / 閾値（平均負荷からの差）	送り手主導（図2の最善解）	受け手主導	送り手主導（相乗り）	受け手主導（相乗り）
1	1	0.82	0.93	0.82
2	1	0.90	0.96	0.91
3	1	0.96	0.99	0.98
4	1	0.97	1.00	0.99
5	1	0.99	0.99	0.98
6	1	1.04	1.01	1.04

情報収集の方針

これまでの解析から高精度の負荷情報が必須であることが判明している．そのため，定期的にプロセス移送グループ内で情報交換を行う．こうした機構はプロセス移送以外の目的にも必要とされることが多い．

プロセス移送の方針

1) 移送判定の主体　受け手主導
2) 移送判定の時期　負荷情報の変化に即応
3) 移送プロセスの選択　残留時間最大のもの．ただし，移送のオーバヘッドを考慮する．
4) 移送先　負荷最小のもの．ただし移送のオーバヘッドを考慮する．

その他

プロセス移送の回数は制限せず，残留処理時間により負荷を評価し，プリエンプションを行う．メッセージの相乗りを行う．

評価例

図2での最善解（送り手主導，最長プロセス選択）に比較し（これを1とする），さらにどれだけ改善がなされたかを示したのが**表 6** である．図2の最善解は既に多くの最適方針を取り入れており，新たな方針は受け手主導とメッセージの相乗りである．受け手主導により，均衡（閾値1）している場合には 20% 近い改善がなされている．重要なことは，図9から明らかなようにプロセス移送の個々の方針の効果が相乗的であることである．

5. 結　　論

プロセス移送は今後普及が期待される分散オペレーティングシステムの重要な機能である．本論文はプロセス移送が十分に効果あるものであり，しかも，実現可能なものであることを示した．

プロセス移送を数学的に解析するのは極めて困難であり，少し条件を変えただけでモデル，解法も異なる．このため，過去に数多く発表されている論文も特定の条件における特定のパラメタの評価となっている．これらのモデルをすべて包含するモデルは理想ではあるが，不可能である．本論文では多くのパラメタを評価するためにいくつかの解析モデルを解析すると共に詳細なシミュレーションモデルを構築し，評価した．解析モデルは主として性能向上の上限を評価するために用い，シミュレーションはより詳細な解析と動的振舞を解析するために用い

た.

これらの結果，過去に提案されている種々の方式を横断的にその最大効果，実際での効果，相互の効果を評価比較することができた．また，本文中に述べたようないくつかの新たな知見も得ることができた．最後に汎用のプロセス移送機構をこれらの結果から示し，それが十分に優れたものであることを示した．

今後，現実の場で使用経験を蓄積し，より良いものへと改善していくことを望んでいる．一方，逆に本論文で得られた観察をより理論的に立証することも行っている．なお，紙数の制限によりモデルの詳細，各グラフ，表の仮定等に関する説明をほとんど割愛せざるをえなかった．さらにはグラフの数等も制限せざるをえなかったため資料が不十分なところも多々ある．

本論文の解析では異なったノードに存在するプロセス間の通信時間やファイルのアクセス時間を考慮に入れてはいない．これは実際のアクセスパタンに基づき解析を行った．これも紙数の都合により省略した．また，性能向上には貢献するもののその効果が小さいために割愛した方針もいくつかある．

これらの点，ご了解をお願いする次第である．

参考文献

1) Barak, A. and Shiloh, A.: A Distributed Load-Balancing Policy for a Multi-Computer, *Softw. Pract. Exper.*, Vol. 15, No. 9, pp. 901-913 (Sep. 1985).
2) Bryant, R. M. and Finkel, R. A.: A Stable Distributed Scheduling Algorithm, *Proc. 2nd Int. Conf. on Distributed Computing System*, pp. 314-323 (Apr. 1981).
3) Casavant, T. L. and Kuhl, H. G.: A Taxonomy of Scheduling in General-Purpose Distributed Computing Systems, *IEEE Trans. Softw. Eng.*, Vol. 14, No. 2, pp. 141-154 (Feb. 1988).
4) Eager, D. L., Lazowska, E. D. and Zahorjan, J.: A Comparison of Receiver-Initiated and Sender-Initiated Adaptive Load Sharing, *Proc. ACM-SIGMETRICS Conf. Measur. and Model. of Computer Systems* (Aug. 1985).
5) Eager, D. L., Lazowska, E. D. and Zahorjan, J.: Adaptive Load Sharing in Homogeneous Distributed Systems, *IEEE Trans. Softw. Eng.*, Vol. SE-12, No. 5, pp. 662-675 (May 1986).
6) Efe, K.: Heuristic Models of Task Assignment Scheduling in Distributed Systems, *Computer*, Vol. 15, pp. 50-56 (June 1982).
7) Finkel, R. A. and Solomon, M.: Distributed Algorithms for Global Structuring, *Proc. the National Computer Conference 48*, pp. 455-460, AFIPS Press (June 1979).
8) Gao, C., Liu, J. W. S. and Railey, M.: Load Balancing Algorithms in Homogeneous Distributed Systems, *1984 Int. Conf. Parallel Proc.*, pp. 302-306 (Aug. 1984).
9) Hac, A. and Johnson, T. J.: A Study of Dynamic Load Balancing in a Distributed System, *Proc. ACM SIGCOMM Symposium on Comm. Architectures and Protocols*, pp. 348-356 (Aug. 1986).
10) Hac, A. and Jin, W.: Dynamic Load Balancing in a Distributed System Using a Decentralized Algorithm, *Proc. 7th Int. Conf. Distributed Computing Systems*, pp. 170-177, IEEE (1987).
11) Hwang, K., Croft, W. J., Goble, G. H., Wah, B. W., Briggs, F. A., Simmons, W. R. and Coates, C. L.: A Unix-Based Local Computer Network with Load Balancing, *Computer*, Vol. 15, pp. 55-66 (Apr. 1982).
12) Kratzer, A. and Hammerstrom, D.: A Study of Load Levelling, *IEEE COMPCOU Fall 80*, pp. 647-654 (Sep. 1980).
13) Lee, K. J. and Towsley, D.: A Comparison of Priority-Based Decentralized Load Balancing Policies, *Proc. Performance '86 and ACM SIGMETRICS*, pp. 70-77 (May 1986).
14) Livny, M. and Melman, M.: Load Balancing in Homogeneous Broadcast Distributed Systems, *Proc. ACM Computer Network Performance Symposium*, pp. 47-55 (Apr. 1982).
15) Lo, V. M.: Task Assignment to Minimize Completion Time, *Proc. 5th Int. Conf. Distributed Computing Systems*, pp. 329-336 (May 1985).
16) Lo, V. M.: Heuristic Algorithms for Task Assignment in Distributed Systems, *Proc. 4th Int. Conf. Distributed Computing Systems*, pp. 30-39, IEEE (1984).
17) Ni, L. M., Xu, C.-W. and Gendreau, T. B.: A Distributed Drafting Algorithm for Load Balancing, *IEEE Trans. Softw. Eng.*, Vol. SE-11, No. 10, pp. 1153-1161 (Oct. 1985).
18) Powell, M. L. and Miller, B. P.: Process Migration in DEMOS/MP, *Proc. 9th Symp. on Operating Systems Principles*, pp. 509-514 (1983).
19) Smith, R. G.: The Contract Net Protocol: High-Level Communication and Control in a Distributed Problem Solver, *IEEE Trans. Comput.*, Vol. C-29, No. 12, pp. 1104-1113 (Dec. 1980).
20) Stankovic, J. A. and Sidhu, I. S.: An Adaptive Bidding Algorithms for Processes, Clusters and Distributed Groups, *Proc. 4th Int. Conf. Distributed Computing Systems*, pp. 49-59,

IEEE (1984).
21) Stankovic, J. A.: Simulations of Three Adaptive, Decentralized Controlled, Job Scheduling Algorithms, *Comput. Networks*, Vol. 8, No. 3, pp. 199-217 (June 1984).
22) Theimer, M. M. and Lantz, K. A.: Finding Idle Machines in a Workstation-based Distributed System, *Proc. 4th Int. Conf. Distributed Computing System*, pp. 112-122, IEEE (1988).
23) Theimer, M. M., Lantz, K. A. and Cheriton, D. R.: Preemptable Remote Execution Facilities for the V-system, *Proc. 10th Symp. on Operating Systems Principles*, pp. 2-12 (1985).
24) Wang, Y. T. and Morris, R. J. T.: Load Sharing in Distributed Systems, *IEEE Trans. Comput.*, Vol. C-34, No. 4, pp. 204-217 (Mar. 1985).
25) Zayas, E. R.: The Use of Copy-On-Reference in a Process Migration System, CMU-CS-87-121 (1987).

付録 動的均衡モデルの解析

n 個のノードに対するポアソン入力と指数サービス時間が与えられ，各ノードに存在するプロセス数の差が $d(d=1,2,\cdots)$ を超えないように移送を行うとする．すなわち，ある瞬間におけるプロセス数最大のノード（高ノードと呼ぶ）とプロセス数最少のノード（低ノードと呼ぶ）のプロセス数の差が d の時，

1) 高ノードの一つに到着があったならば，低ノードの一つに移送する．

2) 低ノードの一つで終了があったならば，高ノードの一つから移送する．

ノード間のプロセスの特定の分布を状態 S と定義すると，S はマルコフ鎖を形成し，定常状態では，任意の瞬間に S が状態 i を取る確率を $P(i)$ と置くと，

$$\Sigma i=o^{m}P(i)=1$$

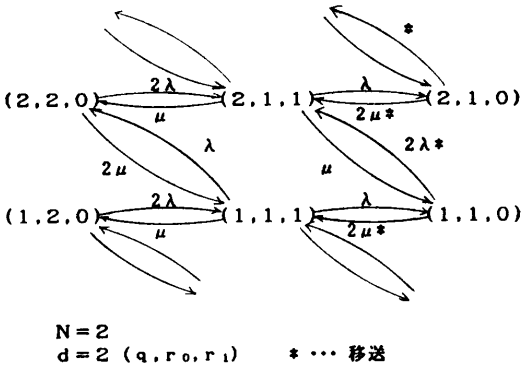


図 A.1 状態遷移
Fig. A.1 State transition diagram.

$$\forall i\ \Sigma_j\theta_{ij}P(i)=\Sigma_j\theta_{ji}P(j)$$

が成り立つ．状態数 m が比較的小さければ，これを数値計算によって解くことができる．

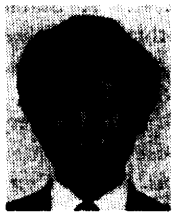
これを解くには状態数を減らすために多くの工夫が必要となるが，ここでは紙数の制限により省略する．

$d=2$ の場合のマルコフ鎖を図 A.1 に示す．

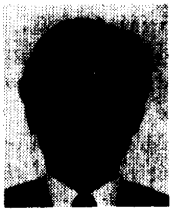
(平成元年8月31日受付)
(平成2年4月17日採録)

朴 圭成
1953年生．1980年大韓民国慶地大学校工科大学電子工学科卒業．1982年同校大学院電子工学科修士課程修了．同年韓国科学技術院システム工学センター入所．1987年東京大学理学系研究科情報科学博士課程入学．1990年韓国科学技術院システム工学センター復職．

芦原 評
1964年生．1987年東京大学理学部情報科学科卒業．1989年同大学院修士課程修了．現在，同博士課程在学中．分散OS，分散アルゴリズムの研究に従事．



清水謙多郎（正会員）
昭和32年生．昭和55年東京大学理学部情報科学科卒業．昭和60年同大学大学院博士課程修了．理学博士．同大学理学部情報科学科助手を経て，平成2年電気通信大学情報工学科講師．LISPマシンの開発，ジョセフソン計算機向きアーキテクチャの設計に従事．現在は分散オペレーティング・システムに関する研究を行っている．電子情報通信学会，日本ソフトウェア科学会，IEEE各会員．



前川 守（正会員）
昭和17年生．昭和40年京都大学工学部数理工学科卒業．東芝を経て，現在東京大学理学部情報科学科助教授．ACM，IEEE，ソフトウェア科学会各会員．