商用大型機での分割区画内のシステム資源の自己管理機能の評価

加倉井 宏一　　荻田 光一郎

日本アイ･ビー･エム･システムズ･エンジニアリング(株)

大規模な商用計算機での分割区画間でのシステム資源を自己管理して最適な稼動を行なう機構の稼働と評価を行う。プロセッサー論理分割環境下でのクラスター構成時、この自己管理機構は、それぞれの論理区画で稼働するワークロードのサービス目標の重要度とCPU資源の競合に応じて、論理区画に割当てられているCPU資源の割合を動的に変更する。あわせて論理区画で割振られる論理CPの数も制御する。

テストとその評価を通じ、この自己管理機能が、バッチジョブのような実行システムを動的に変更できないようなワークロードに対して有効に働き、サービス目標の高いワークロードの稼働する論理区画にCPU資源を割振ることが分かった。また、論理区画へのCPU資源の配分割合に応じて論理CPの数をコントロールし、CPU能力の高い処理を行えることを確認した。

The evalution of a self-optimizing function: Ajusting system resources of logical partitions dynamically

Hirokazu Kakuri　　Kohichiro Ogita

IBM Japan Systems Engineering Co. Ltd.

This paper describe that a self-optimizing function ajust processor resources of logical patitions which are used for a distribution of a processer complex. The function also provide controling the number of logical CPs which are assigned to a logical partition.

The test of the function achive that a dynamic resource re-assignment makes the highest important workload taking CPU resources effectively.

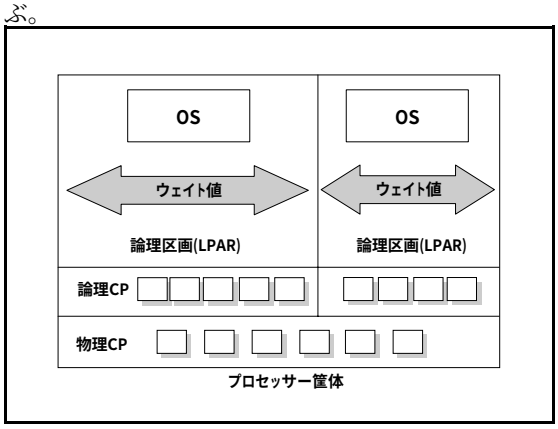
1 はじめに

1.1 論理分割機能

大型商用計算機の分野では、長年、論理分割の機能が使用されている。プロセッサー、メモリー、I/Oチャネルを固定的にしか割振ることのできない物理分割の機能とは異なり、複数オペレーティング･システムをまさしく柔軟な環境で稼動させる機能である。この論理分割機能によって、例えシングルCPのプロセッサーであっても、同一プロセッサー筐体上に本番システムを複数のオペレーティング・システムに分割してソフトウェア障害に対する可用性を高めたり、少ない投資で(つまり新しい機械を購入することなく)開発システムやテストシステムを本番システムから独立させることが可能になった。

現在大型汎用機以外のプラットフォームでも実現され始めている論理分割の機能について簡単に説明する。論理分割機能は、1つの筐体でしかないマシンのリソースを、あたかも複数台のマシンが稼動しているように論理的に分割するハードウェアの機能である。論理的に分割されたマシン・イ

メージを論理区画(LPAR)と呼び、それぞれの論理区画は他の論理区画と完全に独立して稼動できる。分割の対象となるリソースとしては、プロセッサー、プロセッサー・ストレージ、I/Oチャネルがあり、物理的なCPを各論理区画で共有できるのが特徴で、占有でしかCPを使用できない物理分割とは区別される。共用CP使用時は各論理区画にウェイト値という割合を指定することによって、プロセッサー全体のCPU資源を100％使い切った時にその論理区画が使用できるCPU能力の量を設定することができる(図1)。また、論理区画のオペレーティング・システムから見た論理的なCPを論理CPと呼ぶ。



(図1)

1.2 クラスター構成

一方、近年、商業計算機分野においてクラスター構成によるシステム環境が一般化した。クラスター構成とすることにより得られるメリットは、高いスケーラビリティーと可用性である[1]。スケーラビリティーによって、ワークロードの増加、ビジネス規模の増加に応じて、システムスケールを随時増やし、エンドユーザーの処理能力の要求に柔軟に応えることができる。また、同様の処理を行うプロセッサーを複数個用意することにより、例え一つが障害で停止したとしても、エンドユーザーから見て処理が続行可能な可用性を得る事ができる。特に、商用大型機の分野では、そのアーキテクチャー上の強みから、複数システムから同時にデータアクセス可能であり、ディスク装置や共用メモリーを使用したデータ共用によるワークロードバランスが特徴となっている。端末からのトランザクションをフロントエンドシステムで受け付け、システム資源に余裕のあるバックエンドシステムへ処理をルートする。データ共用を行っていることにより、トランザクション処理を行うシステムは、バックエンドのどのシステムでも構わず、ワークロードバランスを考慮したトランザクション処理の負荷分散が可能となっている[2]。負荷分散を行うことにより、良好で均一な応答時間を得ることができ、いつでもエンドユーザーの要求するレベルでのアプリケーションに対する可用性を高めている。

クラスター構成の登場により、データやワークロードを複数システムで共有する機能が提供され、その結果データ共用をサポートし得るアプリケーションはクラスター構成内のどのシステムでも実行できる機会を得ることができるようになった。そうして、システム資源の利用可能な所へワークロードを自由に移動することができるようになった。

1.3 ゴール・オリエンティッドのパフォーマンス管理

さらにオペレーティングシステム内のパフォーマンス管理に目を転じると、ゴール・オリエンティッドのパフォーマンス管理設定では、応答時間や目標の重要度といったユーザーにとって理解しやすい表現方法でサービス目標を設定し、目標達成度を数値で評価することによりシステム全体のスループット向上と管理の容易性を実現することが可能となった[3]。まさしく、ゴール・オリエンティッド(目標指向)なパフォーマンス管理を実現したといえる。具体的にいうと、オペレーティング・システム上で稼動するワークロードに1から5までの重要度を割り振り、さらにそのワークロードに対して応答時間などの目標を定義する。重要度の一番高い指定の重要度１のワークロードから目標を満たすようにオペレーティングシステムがシステム資源を配分し、なおかつシステム全体のスループットを上げようとパフォーマンス管理を動的に行う。従来の定義が複雑なパラメーターでシステム資源の配分を直接コントロールしていたのに対し、ゴール・オリエンティッドのパフォーマンス管理ではユーザーの求めるサービスレベルを直接指定しその指定に沿ってオペレーティング・システムがシステム資源を動的に配分する。

2 自己管理機能

クラスター構成の登場で、データやワークロードを複数システムで共有する機能が提供され、その結果データ共用をサポートし得るアプリケーションはクラスター構成内のどのシステムでも実行できる機会を得ることができた。しかし、技術的に全てのアプリケーションがデータ共用をサポートすることはできず、また、様々な理由によって

データ共用環境に移行できないアプリケーションも数多く存在する。例えば、ネットワーク・リソースの問題でシステムを固定して稼働させなければならないWebフロントエンド・システムやトランザクションのように柔軟に実行システムを変える事ができないバッチジョブなどがその端的な例である。

また、プロセッサー筐体のCPU資源を柔軟に分割できる論理分割機能であるが、CPU資源分割の割合は固定的であり、区画の活動化時に決定される。指定の方法によっては、別の論理区画に処理が無ければ、ウェイト値に関わらずプロセッサー筐体にあるCPU資源を使い切ることができる。しかし、全論理区画が100%のCPU資源を使用する場合、ウェイト値の値は正しく守られる。そのため、計画の段階で各論理区画の処理量からウェイト値を算出するのが重要となるが、Webトランザクションのように前もって十分にトランザクション量を見積もることができない処理の場合ウェイト値の算出が非常に困難となる。

こういった負荷分散に適用できないアプリケーションや、そもそもウェイト値の算出が難しいワークロードののため、分割区画内でのシステム資源の自己管理機能がある。基本的な考え方として、この自己管理機能は、ワークロードの稼動するところにシステム資源を動的移動する機能を実現している、ということができる。動的変更は、論理区画で稼働するワークロードの目標と重要度、そしてCPU資源の充足度から判断して行われる。

分割区画内でのシステム資源の自己管理機能は、2つの管理機能から成り立つ。

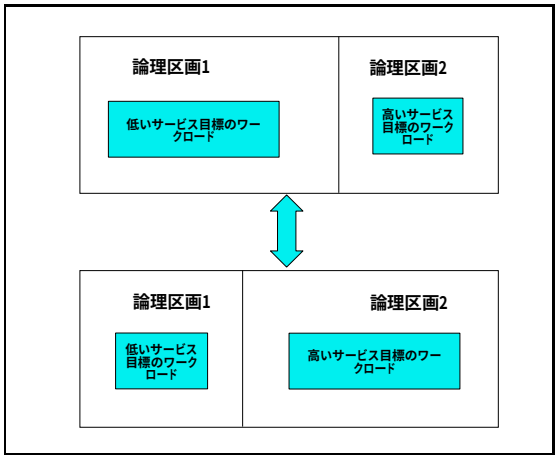
2.1 LPARウェイト管理

LPARウェイト管理は、ワークロードのリソース要求の変化に応じて論理区画のウェイト値を動的に変更することで、サービス目標で指定されたワークロードの目標を満たすためプロセッサー資源をクラスター内で再配分する機能を提供する。重要度の高いワークロードがその目標を満たしていない時、オペレーティングシステムのパフォーマンス管理機能とハードウェア論理分割機能が、そのワークロードが稼動する論理区画のウェイト値を大きくし、プロセッサーの処理能力をその区画に与える(図2)。

ウェイト値の動的変更は次のようにして、決定し実行される。ゴール・オリエンティッドのパフォーマンス管理の場合、稼働するワークロードが目標を達成しているか否かをパフォーマンス・インデックス(PI)という値を元に判断する。PIは次のような式で算出される。

$$PI = \frac{\text{目標実測値}}{\text{目標値}}$$

すなわち、PIが1よりも小さければ目標を満たしており何もしないもしくは他のワークロードに資源を配分し、PIが１よりも大きければ目標を満たしていないのでこのワークロードに配分される資源を増やす動きがなされる。



(図2)

2.2 VARY CPU管理

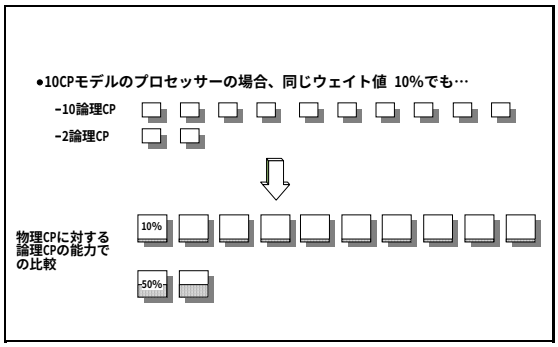
自己管理機能のもう1つの機能として、オペレーティングシステムのパフォーマンス管理機能がそれぞれの論理区画でオンラインになっている論理CPの数を最適化する機能がある。LPARのウェイトの動的変更に伴って論理CPの数を変えることでオペレーティング・システムから見た論理CPの能力とプロセッサーの持つ物理CPの能力の差を限りなく近づけることが可能となる。これをVARY CPU管理と呼ぶ。

2.2.1 論理CPの数の問題点

区画に割当てられるCPU資源の割合は、ウェイト値によって決定される。この割当てられたCPU資源を論理CPで分割して使用することで、区画全体のCPU資源が保証されることになる。このことから、区画に定義されている論理CPの数が多いほど１つの論理CP当たりの能力が低くなってしまう現象が発生する(図3)。

1CPの能力が低い場合発生する問題として、1タ

スクで稼働するオンライン制御プログラムやバッチジョブにおいて、処理に遅延が発生するという問題がある。



(図3)

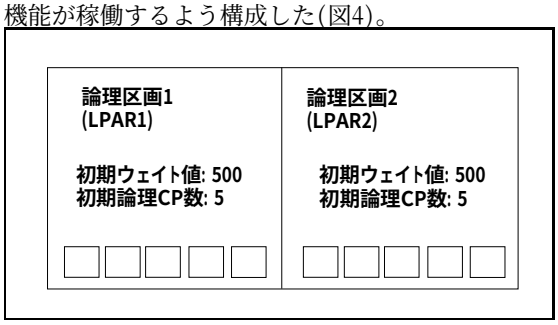
論理CPの数は、ウェイト値に応じてウェイト値に足りる分を確保することで、論理CPの能力を最大にする事になる。また、筐体全体の論理CPの数を少なくすることで、論理分割の処理でのオーバーヘッドを軽減することにもつながる。

3 評価

本章では分割区画内の自己管理機能について評価を行う。

3.1 テスト環境

評価のためのテストを実機を用いて行った。単一筐体での二つの論理区画を準備した。各論理区画で稼働するオペレーティングシステムをクラスター環境で定義を行い、システム資源の自己管理機能が稼働するよう構成した(図4)。



(図4)

テスト用ジョブとしてCPUバウンドのジョブを用意し、高いサービス目標を持つグループと、比較して低いサービス目標を持つグループとで分けた。

本テストでは評価を単純にするため、オペレーティングシステムを除くと、テストとして実行するバッチジョブ以外他のワークロードは稼働させない。

3.2 テスト結果

3.2.1 CPUウェイト管理

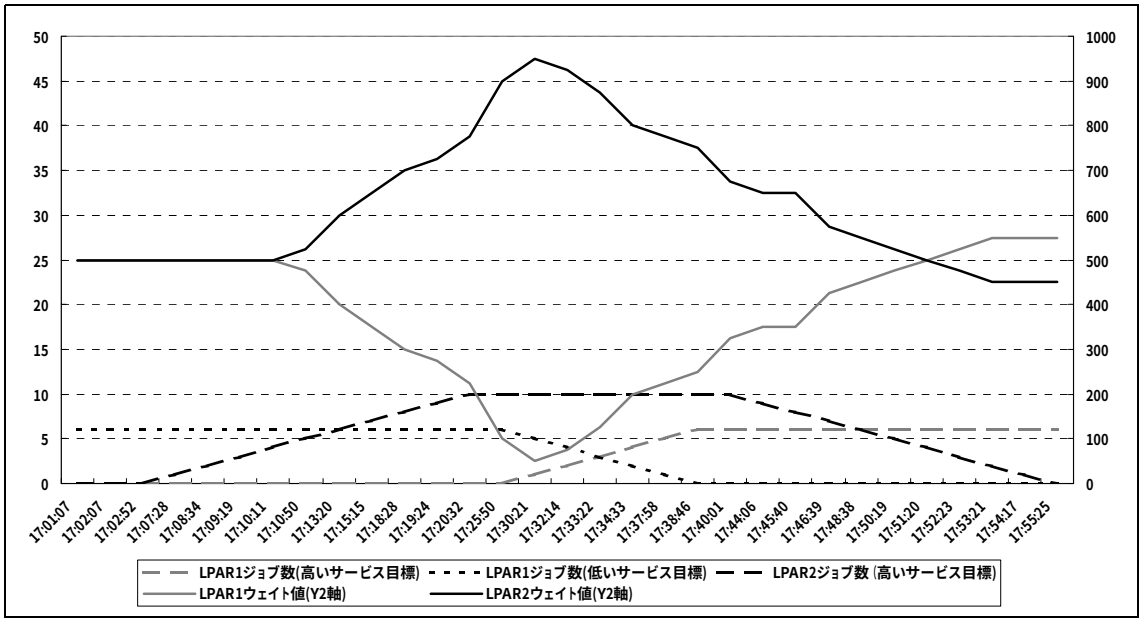
CPUウェイト管理のテストは、低いサービス目標のバッチジョブを片方の区画で稼働させながら、別の区画で高いサービス目標のバッチジョブを稼働させ、CPU資源の競合を発生させる。そして、その時のウェイト値を取得する。

バッチジョブを6本程度片方の論理区画に稼働させただけではシステム資源の競合は発生せず、ウェイト値に変化は見られない(図5 17:02:32)。

さらにテストを続け、LPAR1で低いサービス目標を持ったバッチジョブ6本を稼働させた状態で、順次LPAR2で高いサービス目標を持ったバッチジョブを実行し始める。5本のバッチジョブを稼働させた辺りからシステム競合が発生し、ウェイト値に変化が見られる(図5 17:10:11)。LPAR1で稼働する低いサービス目標を持ったバッチジョブの本数と同数のバッチジョブをLPAR2で高いサービス目標で稼働させた場合、LPAR1とLPAR2のウェイト値は、400:600となり、高いサービス目標を持つバッチジョブの稼働するLPAR2にCPU資源が多く割振られているのが分かる。(図5 17:13:20)

さらに、LPAR2の高いサービス目標を持ったバッチジョブの稼働本数を増加させ、最終的に高い目標を持ったバッチジョブを10本稼働させた時点でLPAR1とLPAR2のウェイト値は、50:950までになった。(図5 17:30:21)

ここからさらに、LPAR1で稼働する低いサービス目標のバッチジョブ群を、高いサービス目標に変更し始める。LPAR1で稼働する高いサービス目標のバッチジョブが増加するにしたがって、LPAR1のウェイト値が回復し始めた。



(図5)

3.2.2 VARY CPU管理

一方、VARY CPU管理についてもテスト結果を評価する。

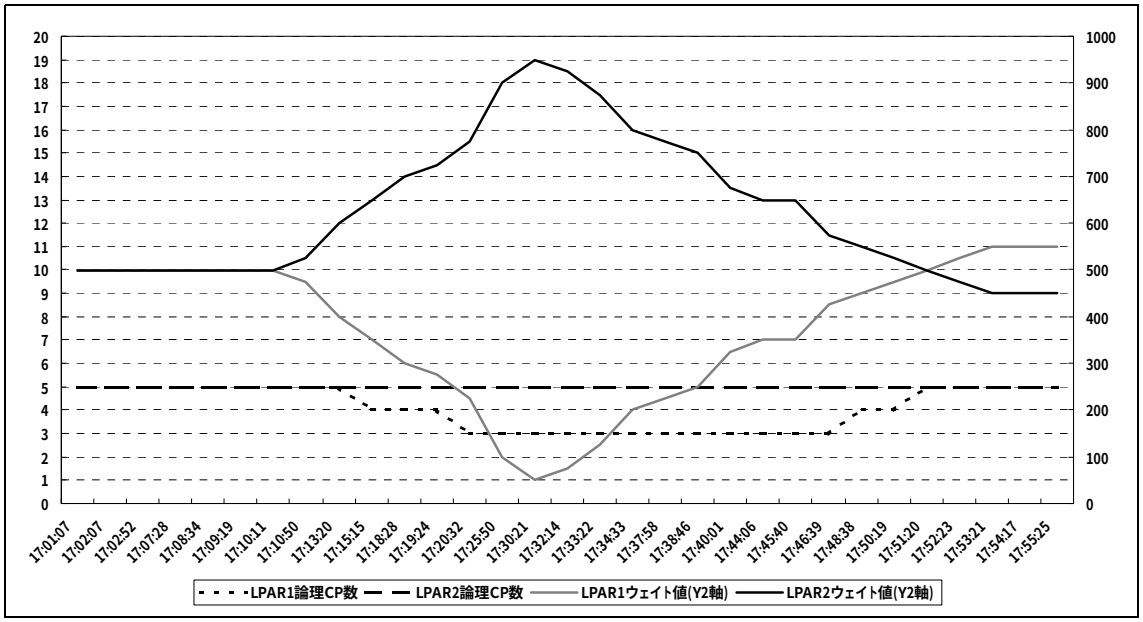
CPUウェイト管理のテストでウェイト値の変化が見られたとき、論理CPの数にどのような変化があったか確認する。

LPAR1とLPAR2とで、最もウェイト値に開きが出た時のウェイト値は、50:950である(図6 17:30:21)。この時の論理CPはそれぞれ3個と5個であった。

論理CPの物理CPに対する能力比は、次のような式で算出することができる。

$$\text{論理CPの物理CPに対する能力(\%)} = \frac{\text{論理区画のウェイト値}}{\text{筐体のウェイト値合計}} \times \frac{\text{筐体の物理CP数}}{\text{区画の論理CP数}} \times 100$$

テスト結果では、LPAR1に注目すると、ウェイト値は50、論理CPは3個で稼働していたため、論理CP1つの能力は、物理CPと比較して8.4%である。もし仮に、論理CPが5個であった場合、物理CPの能力の5%になるため、論理CP数が減少したことにより、1.7倍のCPの能力を発揮できることになる。



(図6)

4 まとめ

以上のように、自己管理機能について一定の評価を得ることができた。まとめとして、今回の評価のポイントと課題、今後の評価について探っていきたい。

本研究では、論理分割区画内でのシステム資源の自己管理機能が有効に働くことを評価できた。論理区画内で稼働するワークロードの重要度に応じて、CPU資源の分割が動的に調整されるのを確認した。また、CPU資源の分割の割合に応じて論理CPの数が動的に調整され、1CPの能力が高くなるように稼働することを確認した。

本評価での課題は限定的な利用、すなわちテスト環境として2つの論理区画と2つの重要度のグループで分けたバッチジョブ群を利用したことにあると考える。

実際のお客様の稼働環境としては、もっと複雑な構成となる。論理区画数も多く、重要度も2つよりも多い。また、ワークロードとしてバッチジョブだけでなく、オンライントランザクション処理、そして最近であれば、ビジネス･インテリジェンスと呼ばれるような経営計算処理が含まれるであろう。

また、データ共用に関する点も本評価では除外している。データ共用環境の場合考えられる課題として、ある区画で稼働するワークロードが別の区画で稼働するワークロードと関連するため、論理区画のCPU資源を単純に割振っただけではクラスター全体で処理効率が悪くなる可能性がある。

今後、本研究よりもっと複雑な構成での、評価、研究を進めていく必要があると考えている。

参考文献

[1]J. M. Nick, B. B. Moore, J.-Y. Chung, and N. S. Bowen、"S/390 cluster technology: Parallel Sysplex"、IBM Systems Journal, Vol.36, No.2, 1997

[2]T. Banks, K. E. Davies, and C. Moxey、"The evolution of CICS/ESA in the sysplex environment"、IBM Systems Journal, Vol.36, No.2, 1997

[3]J. Aman, C. K. Eilert, D. Emmes, P. Yocom, and D. Dillenberger、"Adaptive algorithms for managing a distributed data processing workload"、IBM Systems Journal, Vol.36, No.2, 1997