ハイブリッドクラウドの各種コストを考慮したデータ処理負荷分散ミドルウェアの提案

笠江優美子† 小口 正人†

† お茶の水女子大学 〒 112-8610 東京都文京区大塚 2-1-1
E-mail: †yumiko@ogl.is.ocha.ac.jp, ††oguchi@computer.org

あらまし 近年，コンピュータシステムにおける情報量が爆発的に増大しており，そのデータを効率よく処理するシステムが求められている．本研究では，そのようなシステムをハイブリッドクラウド環境において実現するミドルウェアを提案する．本ミドルウェアは，ハイブリッドクラウド環境において，大量のデータを効率よく処理するとともに，パラメータの設定により省電力指向を含む金銭的コストを抑える負荷分散を実現する．さらに本論文では，提案するミドルウェア実行時の処理時間，従量制料金，消費電力量料金を元に，ミドルウェア実行時のトータルコストを評価することで，本ミドルウェアの有用性を示した．

キーワード ハイブリッドクラウド，消費電力量，ミドルウェア，負荷分散，コスト評価，データインテンシブアプリケーション

Proposing the Method for Data Processing Load Distribution Considering the Various Costs on the Hybrid Cloud

Yumiko KASAE† and Masato OGUCHI†

† Ochanomizu University 2-1-1 Otsuka, Bunkyouku Tokyo 112-8610 JAPAN
E-mail: †yumiko@ogl.is.ocha.ac.jp, ††oguchi@computer.org

1. はじめに

近年，コンピュータシステムにおける情報量の爆発的増加により，大量のデータを効率よく処理できるシステムが求められている．そのシステムを実現する手段として，クラウドコンピューティングが期待され，世界中で普及してる．しかし，その普及に伴いクラウドを構築する側が持つサーバなどのコンピュータ機器の消費電力量の増加が懸念されている．さらに，世界的なエコ志向により，それらコンピュータ機器の消費電力量削減も求められている．消費電力量を削減する手段として，空調機器やコンピュータ機器自身の省電力化も考えられるが，このようなハードウェアからの取組みは，コストの観点からも必ずしも容易ではない．そのためソフトウェアの観点からの省電力化が有効である．そこで本研究では，パブリッククラウドとプライベートクラウドを併用するハイブリッドクラウド環境において，大量のデータを効率よく処理し，消費電力を含む金銭的コストを抑える負荷分散を行うミドルウェアを提案する．

本ミドルウェアの特徴は2つある．まず1つめの特徴として，データインテンシブなジョブを対象としている点である．そのため，負荷分散先のすべてのマシンの Disk I/O を定期的に測定し，それを負荷分散の判断材料としている．2 つめの特徴として，消費電力量を含む金銭的コストを抑える負荷分散を行える点である．このミドルウェアでは，ジョブの実行時間という時間的コスト，パブリッククラウドの従量制料金とプライベートクラウドの消費電力量料金という金銭的コストの 2 つのコストを考えている．ここでパブリッククラウドの消費電力量料金については，一般的にその料金がわかるということは考えづらいため，従量制料金に含まれると考えいている．本ミドルウェアでは，この金銭的コストを抑えることで，省電力指向な負荷分散も行える．ユーザは，ミドルウェアのパラメータを変化させることで，ジョブの実行時間を重視した負荷分散にするか，省電力指向を含む金銭的コストを重視した負荷分散にするか，もしくはその中間となる負荷分散を行うかを選択できる．

本論文では，そのミドルウェアを実行させた時の，ジョブの実行時間，パブリッククラウドの従量制料金，プライベートクラウドの消費電力量料金を測定，算出し，それらをもとに時間的コストと金銭的コストを考えた．さらに，ユーザがそれら 2 つのコストの重要度を変化させた時のコスト評価を行った．

2. 関連研究

クラウドコンピューティングにおけるロードバランスを議論している論文として，例えば [7] や [8] などがある．しかしこれらの論文では CPU インテンシブなアプリケーションを対象負荷としており，データインテンシブアプリケーションは考慮していない．ある種の科学技術計算のように計算処理が中心となるアプリケーションの場合は，各ノードの CPU 負荷に基づいて判断し適切な負荷分散を行うことができるが，本研究のようにデータインテンシブアプリケーションがジョブの対象の場合，CPU は I/O 待ちとなっていることが多く，判断基準として使うことは難しい．そこで本研究では負荷の指標として，ディスクアクセス量を用いる．

また，本研究と同じように，ジョブの対象をデータインテンシブアプリケーションとし，その負荷の指標としてディスクアクセス量を用いて開発された負荷分散ミドルウェアに [5] がある．このミドルウェアでは，ディスクアクセス量を元に，ローカルのクラスタとパブリッククラウド間で動的な負荷分散を行うことで，ジョブの最適配置を行った．本研究ではこのミドルウェアを更に発展させ，ユーザによるパラメータ設定により，省電力指向を含む金銭的コストを抑える負荷分散も可能である．

さらに，クラウドクラウドコンピューティングの省電力化の検討も活発にされている．[11] と [10] などは，本研究とは異なり，CPU インテンシブなアプリケーションをクラウド上で実行するときの省電力化への取組みである．[9] は，クラウドのデータセンタに対する省電力化への取組みである．本研究とは，プライベートクラウドを含むクラウド全体の省電力化を目指す点で異なる．[13] は，消費電力量とジョブの実行時間を考慮したスケジュールリングアルゴリズムを提案している．しかし，本研究では，ハイブリッドクラウド環境において，データインテンシブなアプリケーションに対して，そのジョブの実行時間と，パブリッククラウドの従量制料金とプライベートクラウドの消費電力量をコストとして考え，それを抑えようとしており，その点でこれらの研究とは異なる．

3. ミドルウェア実行環境

本研究では，クラウド構築ソフトウェア Eucalyptus [1] を用いて 2 つのクラウドを構築した．Eucalyptus では，仮想化ソフトウェアとして，Xen [2] と KVM [3] の 2 種類をサポートしている．先行研究 [6] では，Eucalyptus で構築したクラウドにおいて，データインテンシブなジョブを実行させる場合，KVM より Xen の方が仮想化ソフトウェアの特性からも優れていることがわかった．そのため，仮想化ソフトウェアとしては Xen を採用し，実験システムを構築する．Eucalyptus は，Cloud controller(CLC) , Cluster controller(CC) , Node controller(NC) の 3 層構造である．CLC では EC2 互換のインターフェイスを備え，ユーザから指示された内容に基づいた各種の制御を行い，CC ではインスタンス間，インスタンスとクラウドの外部の間のネットワークの制御を行う．NC は実際にインスタンスを動作させる．インスタンスとは，クラウドから提供される仮想マシンである．

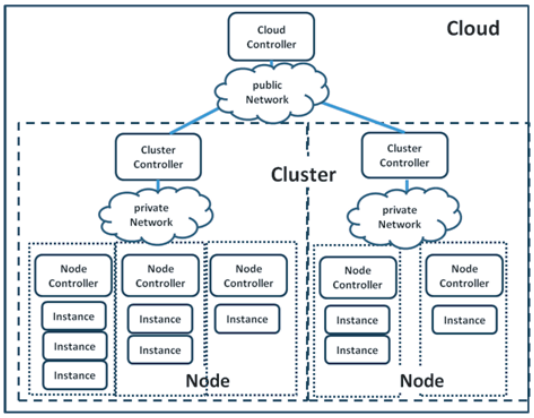

図 1　Eucalyptus のアーキテクチャ

図 2 に本ミドルウェアの実行環境を示す．プライベートクラウドとして，CC と CLC が動作する Frontend サーバ 1 台とインスタンスが生成される Node サーバ 4 台，パブリッククラウドも同様に，Frontend サーバ 1 台と Node サーバ 4 台とし，その間を人工的に遅延を発生させる dummynet で繋いだ．ハイブリッドクラウド環境として，それぞれの性能は表 1 から表 4 に示す通りである．2 台の Frontend サーバはネットワークの口を 2 つ持ち，1 つが Node サーバへ，もう 1 つが外部のネットワークへ繋がっており，これによってそれぞれの Node サーバは外部のネットワークから独立している．

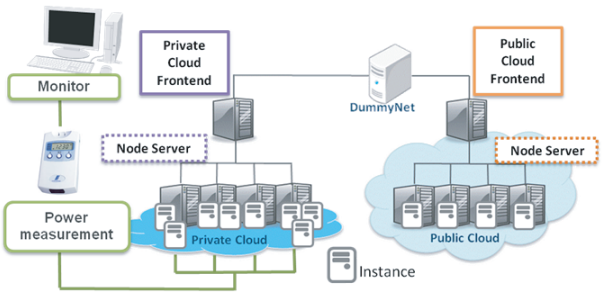

図 2　構築したハイブリッドクラウド環境

表 1　Private Cloud Frontend

OS	Linux 2.6.38/Debian GNU/Linux 6.0
CPU	Intel(R) Xeon(R) CPU @ 3.60GHz
Memory	4GByte
Disk	141GByte

表 2　Private Cloud Node

OS	Linux 2.6.32-xen-amd64 and xen-4.0-amd64 /Debian GNU/Linux 6.0
CPU	Intel(R) Xeon(R) CPU @ 3.60GHz
Memory	4GByte
Disk	222GByte

この環境で電力を測定するための測定機として，システムアー

表 3　Public Cloud Frontend

OS	Linux 2.6.38/Debian GNU/Linux 6.0
CPU	Intel(R) Xeon(R) CPU @ 2.40GHz
Memory	1GByte
Disk	72GByte

表 4　Public Cloud Node

OS	Linux 2.6.32-xen-amd64 and xen-4.0-amd64 /Debian GNU/Linux 6.0
CPU	Intel(R) Xeon(R) CPU @ 3.60GHz
Memory	4GByte
Disk	141GByte

トウェア製の高精度小型電力計ワットアワーメータ SHW3A [4] を用いた．これは，コンセントに接続したワットアワーメータに電気製品を繋ぐと，瞬時に消費電力を測定，表示するというものである．本研究では，電力の測定対象はプライベートクラウドのみとし，プライベートクラウドの Node サーバが消費する電力を測定する．これは，現実としてパブリッククラウド環境で消費電力量がわかることは想定しづらいためである．その代わりパブリッククラウドにおいては，従量制料金のコストがかかるものとする．これについては後述する．測定された消費電力量は，電力モニタ用 PC で収集される．

4. 提案する負荷分散ミドルウェア

4.1 ミドルウェア概要

本節に，提案するミドルウェアの概要を示す．本ミドルウェアでは，データインテンシブアプリケーション実行時に，プライベートクラウドとパブリッククラウドの Disk I/O をモニタリングすることで実行されているジョブ量を推定し，投入されたジョブの負荷分散を行う．すなわち，データインテンシブアプリケーションのジョブが連続的に投入されている状況において，プライベートクラウドのリソースを使い切ったら，パブリッククラウドへ負荷分散を行う．パウブリッククラウドにおいても，ジョブが投入されたインスタンスのリソースを使い切ったら，新たにインスタンスを借り，ジョブを投入していく．このように，リソースを使い切ってから負荷分散を行なっていくことで，後述するミドルウェアの評価指標として考えるコストを抑えていき，効率よくジョブを処理する．さらに，ユーザの指示によりパラメータ設定を変えることで，消費電力量を含む金銭的コストを抑える負荷分散も実現する．

本ミドルウェアはプライベートクラウドの Frontend 上で動作しているシェルスクリプトと C 言語のプログラムで，ローカルクラスタおよびクラウドリソースの Disk I/O を測定する Monitor 部とジョブを振り分ける Dispatch 部から成る．Monitor 部では，dstat コマンドを利用して定期的に Disk I/O の情報を収集している．一方 Dispatch 部では，投入されたジョブを受け取り，Moitor 部で収集した Disk I/O の情報を基に，プライベートクラウドまたはパブリッククラウドへジョブを負荷分散する．

4.2 飽 和 判 断

本ミドルウェアでは，リソースを使いきるための判断として，Disk I/O の値を用いている．これは，ジョブとしてデータインテンシブなジョブを使用してるためである．データインテンシブなジョブの場合，処理が I/O 待ちとなっていることが多く，CPU 負荷による判断が難しいため，Disk I/O から判断を行う．図 3 に，データインテンシブなジョブの投入量を増やしていった場合の Disk I/O と実行時間のグラフを示す．

この図 3 から，ジョブの投入量が増加すると，Disk I/O の値は飽和状態となり，この状態になると実行時間は飽和していない状態を表した baseline に場合に比べ，長くなっていってしまうことが読み取れる．baseline は，データインテンシブなジョブを横軸の個数分のジョブを並列に処理させるのではなく，直列に処理させた時の処理時間である．つまり，1 つのジョブが終わったあとすぐに，次のジョブを投入していく場合の処理時間を意味する．本ミドルウェアでは，リソースが飽和すると負荷分散を行う．そのために，ミドルウェアでは，定期的に Disk I/O の値を測定し，飽和状態の Disk I/O の値 (以下，S 値とする) を任意回測定したら，別のインスタンスへ負荷分散を行い，S 値を任意回下回ったら，そのインスタンスは飽和していないと判断している．Disk I/O の値は不安定であることからも，上記方法は飽和状態を見積もる妥当な方法の 1 つであると考える．

本ミドルウェアでは，ユーザがこの飽和判断レベルを変化させることで，ジョブの処理時間を重視した負荷分散を行うか，省電力指向を含む金銭的コスト重視な負荷分散を行うかを選択できる．この飽和判断レベルを変化させるとは,「インスタンスの Disk I/O の値が S 値を超えた回数（以下，U 値とする)」と「インスタンスの Disk I/O 値が S 値を下回った回数（以下，L 値とする)」を変化させることである．例えば，U 値を下げ，L 値を上げれば，ジョブは負荷分散されやすくなり，パブリッククラウドが多用される．つまり，金銭的コストが多くなるが，ジョブの処理時間は早くなる．逆に，U 値を下げ，L 値を上げれば，ジョブは負荷分散されにくくなり，ほとんどのジョブはプライベートクラウド内で処理される．この場合は，ジョブの処理時間は長くなるが，消費電力量料金を含む金銭的コストを下げることができる．

4.3 ミドルウェアの評価指標

クラウドに負荷分散することを考えた場合，パブリッククラウド側で多くのインスタンスを用い，多くの処理を並列して実行すれば，全体の実行時間が短くなることが期待される．しかしパブリッククラウドは従量制のコストが発生するため，実行時間のみを考慮した場合，コストが膨大になる可能性がある．さらに，パブリッククラウドを多用すれば，プライベートクラウドの消費電力量料金は抑えられるが，パブリッククラウドの従量制コストがかかってしまう．従ってパブリッククラウドを使用した負荷分散システムを実現し，さらに消費電力料金を含む金銭的コストを抑えるような負荷分散を目指す場合は，実行時間などのパフォーマンスとともに，従量制コストや消費電力量も考慮してリソースを配分することが重要である．そこで本ミドルウェアの評価指標として，ジョブの実行時間という時間

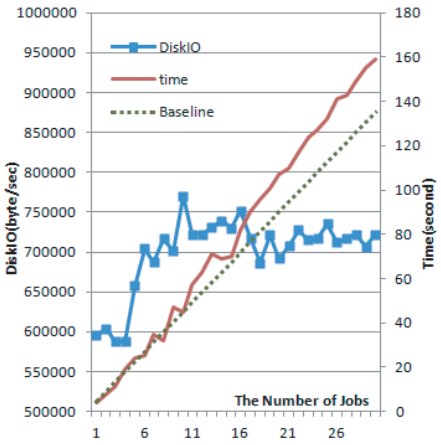

図 3　データインテンシブアプリケーション実行時の Disk I/O と処理時間

的コストと，パブリッククラウドの従量制料金とプライベートクラウドの消費電力量料金を合わせた金銭的コストを考える．

4.4　ミドルウェアのアルゴリズム

本実験において，ジョブは連続的に投入されていることを想定し，以下のアルゴリズムをミドルウェアに実装した．

(1)　連続投入されたジョブを受け取る．

(2)　プライベートクラウドにおいて実行優先順位が高いインスタンスから順次飽和しているか調べ，飽和していなければプライベートクラウド内のインスタンスで実行して (1) へ．飽和していれば (3) へ．

(3)　パブリッククラウドにおいて実行優先順位が高いインスタンスから順次飽和しているか調べ，飽和していないものが見つかり次第実行して (1) へ．見つからなければ (4) へ．

(4)　パブリッククラウドにおいて借りるインスタンスを増やし，実行して (1) へ．

まずはプライベートクラウド，次はパブリッククラウドの優先順位が高いものから Disk I/O の状態を調べ，空いているところから実行を行うことで，時間的コストと金銭的コストの両方がバランス良くなるように考慮している．

5.　ミドルウェア実行実験と測定結果

5.1　ミドルウェア実行実験概要

本ミドルウェアを実行させる際，実験では，図 4 に示すように，どちらのクラウドにおいても，Node サーバに表 5 の性能のインスタンスを 1 つずつ生成し，合計 8 つのインスタンスに対し，負荷分散を行なった．パブリッククラウドでは，インスタンス数は制限が無く利用できることが一般的であると考えられるが，本実験で投入するジョブの総量は，パブリッククラウド 4 つ以内で十分に負荷分散可能なものである．

実験では，第 4 章で述べた L 値を固定し，U 値を表 6 のように飽和判断レベルを変化させた．この飽和判断レベルでは，小さいほど負荷分散されやすく，大きいほど負荷分散されづらいことを意味する．

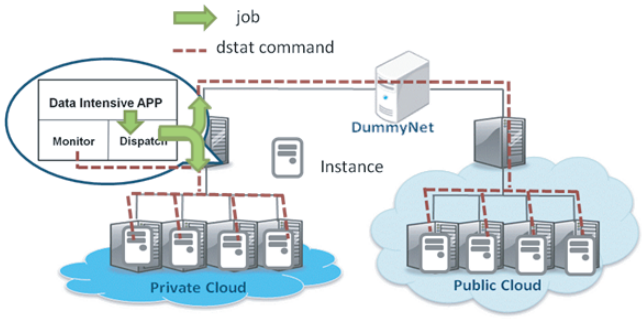

図 4　実験システム

表 5　Instance

OS	Linux 2.6.27.21-0.1-xen / x86_ 84 GNU / CentOS 5.3
CPU	Intel(R) Xeon(R) CPU @ 3.60GHz 1 core
Memory	1024MByte
Disk	10GByte

表 6　飽和判断レベル

Level	L 値	U 値
I	5	2
II	5	3
III	5	4
IV	5	5
V	5	6
VI	5	7
VII	5	8
VIII	5	9
IX	5	10

このように，飽和判断レベルを変化させたときの，ジョブの処理時間，パブリッククラウドの従量制料金，プライベートクラウドの消費電力量料金を測定し，その評価を行う．

5.2　データ配置

クラウドを利用した負荷分散の際には，データ配置を考えることも重要である．その配置として，クラウドに付随するブロックストレージを使う場合や，ローカルにストレージを設けてパブリッククラウドからは遠隔アクセスを行う場合などが考えられるが，本実験では，遠隔バックアップなどにより，インスタンス内に既にデータが配置されている場合を想定し，実験を行っている．今後上記データ配置についても取り組んでいきたい．

5.3　投入するジョブ

本実験で投入するジョブは，データインテンシブなアプリケーションを想定するため，PostgreSQL のベンチマークである pgbench を用いた．pgbench は PostgreSQL に同梱されているシンプルなベンチマークツールである．最初のバージョンは石井 達夫氏により作成され，日本の PostgreSQL メーリングリストで 1999 年に公開された．その後 contrib という付属追加プログラムとして，PostgreSQL のソースコードとともに配布されるようになった．サーバ側データベースの基本的なベンチ

マークに用いられる TPC-B を基に作成されており，1 秒間に実行できるトランザクション数で性能を判断可能である．本実験では，すべてのインスタンスのローカルに PostgreSQL を用いて 6Gbyte のデータベースを作成し，client 数 1, transaction 数 500 のジョブを 2 秒間隔で 200 回投入した．1 ジョブあたり処理時間は，平均 9 秒程度である．今回の実験では，pgbench のように細かい独立したデータインテンシブなジョブが，次々と投入されていく場合を想定している．

5.4 測定結果

図 5，6，7 に，飽和判断レベルを変化させたミドルウェア実行時のジョブの処理時間と，パブリッククラウドの従量制料金，プライベートクラウドの消費電力量の測定結果を示す．どのグラフも横軸は，投入したジョブ回数を示している．図 5 は，投入したジョブ回数までのジョブすべてが終わるまでの時間を示している．図 6 は，パブリッククラウドに投入したジョブの実行時間とパブリッククラウドから借りたインスタンス台数，パブリッククラウドの従量制料金を掛け，算出したものである．パブリッククラウドの従量制料金は，代表的なクラウドプロバイダである，AmazonEC2 の従量制料金を元に，本実験環境のインスタンス性能を考え，1 時間あたり$0.5 として計算した．図 7 は，図 5 と同様，投入したジョブ回数までのジョブすべてが終わるまでの消費電力量料金を示している．消費電力量料金は，東京電力の電気料金を参考に，1 kWh あたり$0.5 として計算した．これら 3 つのグラフを見てみると，図 5 においては，level I と level II の処理時間が等しいにも関わらず，図 6 では，level I のほうが level II に比べ，従量制料金が高くなっている．これは，今回投入したジョブのすべては，level II までの状態で十分に負荷分散可能なジョブであり，level I では，無駄にパブリッククラウドのリソースを使ってしまっているためだと考えられる．level III から level IX においては，level が上がるに連れ負荷分散されにくくなっていることが，図 6 の従量制料金が低くなっている点から読み取れる．それに伴って，ジョブの処理時間もかかっていることがわかる．また，図 7 は，図 5 と形状が似ていることから，消費電力量は処理時間が多大な影響を与えていることがわかる．このように，本ミドルウェアで飽和判断レベルを変化させることで，ジョブの実行時間と，パブリッククラウドの従量制料金，プライベートクラウドの消費電力量料金を制御できる．

また，図 8 に，本ミドルウェアを実行させた時の，各インスタンスの Disk I/O の値とジョブ単位あたりの実行時間を示す．まず，プライベートクラウド 1 と名付けられたインスタンスにジョブが投入されていき，このインスタンスが Disk I/O により飽和していると判断されたら，プライベートクラウド 2 にジョブが投入される．プライベートクラウド 2 が飽和したら，プライベートクラウド 3 へ，プライベートクラウド 3 が飽和したら，プライベートクラウド 4 へ負荷分散していることが読み取れる．そして，これまでの間に最初にジョブを投入したプライベートクラウド 1 の飽和状態が緩和されるため，次のジョブはプライベートクラウド 1 へ投入される．この状態になると，プライベートクラウド内のすべてのインスタンスが飽和状態となってしまったため，次のジョブはパブリッククラウドへ負荷分散されている．このように，本ミドルウェアにより Disk I/O による負荷分散の制御が実現できている．

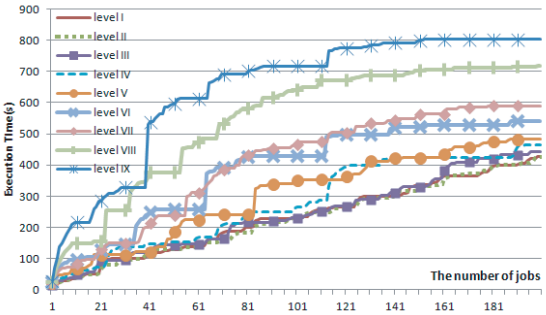

図 5　実行時間

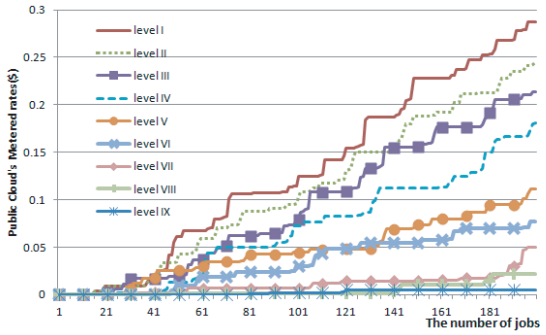

図 6　パブリッククラウドの従量制料金

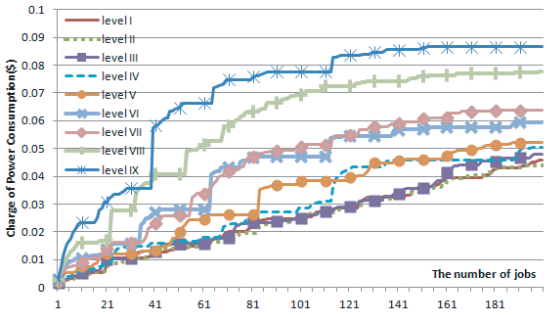

図 7　プライベートクラウドの消費電力量料金

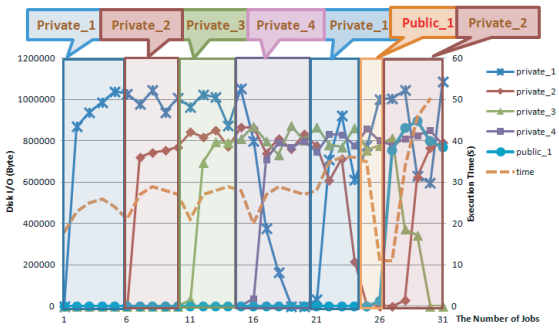

図 8　ミドルウェア実行時の Disk I/O の推移とジョブ単位あたりの実行時間

6. ミドルウェアの評価

6.1 評価概要

本章では，第 4 章にて測定した結果をもとに，本ミドルウェアの評価を行う．本ミドルウェアでは，評価指標として以下の式を考える．

Total cost = F* T_{total}+ (T_R * N_R * C_R +P_L * C_L)

T_{total}:Execution time of Total Jobs
T_R:Execution time of Public Cloud
N_R:Number of Instances used on Public Cloud
C_R:Charges of Public Cloud
P_L:Power Consumption on Private Cloud
C_L:Charges of Power Consumption on Private Cloud
F:The Factor for Convert Money into Time

第一項が実行時間という時間的コスト，第二項がパブリッククラウドの従量制料金とプライベートクラウドの利用時間分の消費電力量料金を合わせた金銭的コストを表す．パブリッククラウドの消費電力量料金は，パブリッククラウドの従量制コストに含まれると考える．これは，現実としてパブリッククラウドの消費電力量料金がわかるということは想定しづらく，また分かったとしてもその電力を削減しようとするかどうかはユーザのモチベーションに依存するためである．係数 F においては，ユーザが時間的コストと金銭的コストのバランスをどのように考えるかを決める値であり，時間的コストを金銭的コストに換算するための係数である．本章では，まず従量制料金と消費電力料金の価格設定による金銭的コストの議論を行う．そして，定めた金銭的コストを基に，上記式の定数 F の比率を変化させたトータルコストによる評価を行う．

6.2 金銭的コストに関する議論

本研究では，金銭的コストをパブリッククラウドの従量制料金とプライベートクラウドの消費電力量料金で考える．このことに関する議論として，この 2 つの金額は常に一定ではないことが挙げられる．最近はクラウドを提供するプロバイダが数多く存在しているが，その従量制料金は必ずしも一定ではなく，今後このようなプロバイダが更に増えれば，従量制料金は価格競争により下がるかもしれないし，何らかの事情により，現在より高くなるかもしれない．これは，消費電力量料金においても言えることである．これらのことを踏まえ，本実験の評価では測定した従量制料金とプライベートクラウドの消費電力量料金の価格設定を変化さえた様々な金銭的コストを考える．実験では，1 インスタンスあたりの 1 時間毎の従量制料金を\$0.5 ，1kWh あたりの消費電力量料金を\$0.5 と換算し測定したが，本評価では図 9 にあるように，従量制料金を\$0.5 から\$1.5 ，消費電力量料金を\$0.5 から\$3.0 まで変化させた．一方を変化させるときは，もう一方は初期値で固定とした．

図 9 から，ほとんどの価格設定において，パブリッククラウドを最も多用し，従量制料金が高くなる level I の金銭的コストが大きくなっている．これは，本実験で投入したジョブの総量では，パブリッククラウドの従量制料金の方が，プライベートクラウドの消費電力量料金に比べ割高になっているためであると考えられる．しかし，消費電力量料金を高額な\$3.0 に変化させると，どの level でも比較的均等な金額となる．これは，低い level においては，従量制料金が金銭的コストの大半を占め，level が大きくなるに連れ，従量制料金は小さくなるが，プライベートクラウドの消費電力量料金の割合が増えて，結局のところ金銭的コストの総和は変わっていない状態である．これらの結果から，図 9 の中で，象徴的な結果である，消費電力料金:従量制コスト=\$3.0：\$0.5 ，\$1.5：\$0.5 ，\$0.5：\$0.5 ，\$0.5：\$1.0 ，\$0.5：\$1.5 の結果を金銭的コストの代表例とし，これらの金銭的コストを用いてトータルコストの評価を行う．

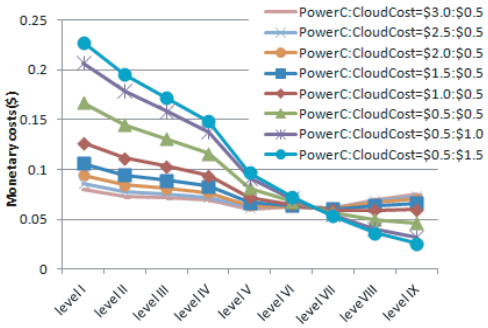

図 9　金銭的コスト

6.3 トータルコストによる評価

トータルコストの評価では，6.1 節で示した式を用いるが，係数 F が重要となる．係数 F は，ミドルウェアの全体のジョブの実行時間を金銭的コストに換算する係数である．F の値が大きい場合は，時間的コストを重要視し，小さい場合は金銭的コストを重要視する．今回の評価では，その値の代表例として 1/20，1/200，1/2000，1/20000 と変化させた．つまり，1/20 では，処理時間を最も重要視しており，1/20000 は，パブリッククラウドの従量制料金とプライベートクラウドの消費電力量料金による金銭的コストを最も重要視する．図 10，11，12，13 に，係数を順に変化させたときのトータルコストを示す．また，図では，5.2 節で議論したように，金銭的コストの価格設定を変化させている．

図 10 は，最も処理時間を重要視した結果であるため，各価格設定の変化が殆ど無い．この結果はほとんど処理時間のみに依存する結果であるといえる．このことから，最も飽和判断レベルの低い level 1 にユーザがパラメータを設定することにより処理時間重視の負荷分散を提供する，妥当な結果となっている．図 11 は，図 10 に比べると，金銭的コストを少しは考慮しているが，違いは殆ど無い．図 10 と同様に，最も飽和判断レベルの低い level 1 をユーザが指定することで，トータルコストが抑えられる．

しかし，図 12 になると様子が異なる．これは，ある程度処理時間を考慮しつつ，金銭的コストも抑えたい場合のコスト評価である．まず，価格設定により，トータルコストの差が生まれていることがわかる．消費電力料金:従量制コスト=\$3.0：\$0.5 ，\$1.5：\$0.5 のような，プライベートクラウドの消費電力量料

金を重視した価格設定の金銭的コストの場合は，level I から V にかけて，殆どトータルコストの違いがないく，これらのコストが最も小さい．つまり，今回実験のジョブ量では，時間的コストや金銭的コストを考えたトータルコストでは，level V までの間では負荷分散させようとさせまいと，ほとんど違いはなく，またそのコストが最小であるということがわかる．

このことから，この価格設定においては，本ミドルウェアによって，level V までのパラメータをユーザが設定することで，時間的コストと金銭的コストをある程度抑えた負荷分散を提供できることがわかる．また，消費電力料金:従量制コスト=$0.5 : $0.5 , $0.5 : $1.0 , $0.5 : 1.5$ のような，プライベートクラウドの消費電力量料金は固定で，従量制料金が変化する価格設定の場合は，低い level でのトータルコストは大きいが，level V か leve VI の間で最小となり，これ以降だとまたトータルコストが大きくなる．この結果から，重量制料金を重要視した価格設定の場合のトータルコストを考えると，本ミドルウェアによって，level V か level VI のパラメータをユーザが設定することで，時間的コストと金銭的コストの両方をある程度抑えた負荷分散を提供できることがわかる．

図 13 では，時間的コストをほとんど考慮しないため，金銭的コストの影響が大きく表れており，金銭的コストを重視した場合のトータルコストである．消費電力料金:従量制コスト=$3.0 : $0.5 , $1.5 : $0.5 のような，プライベートクラウドの消費電力量料金を重視した価格設定の金銭的コストの場合は，どの飽和判断レベルでも，トータルコストに違いは殆ど無い．これは，このような価格設定で，かつ金銭的コストを最重要視する場合においては，ユーザがどのようなパラメータ設定を行なってもトータルコストは変わらないことがわかる．また，消費電力料金:従量制コスト=$0.5 : $0.5 , $0.5 : $1.0 , $0.5 : $1.5 のような，プライベートクラウドの消費電力量料金は固定で，従量制料金が変化する価格設定の場合は，level IX のようにほとんどパブリッククラウドを使わず，プライベートクラウド内のリソースでジョブを実行させた方がコスト的に良いことがわかる．そのため，ユーザが，このような金銭的コストの価格設定において，金銭的コストを時間的コストより抑えたいと言う場合には，level IX のようなパラメータに設定するば良いことがわかる．

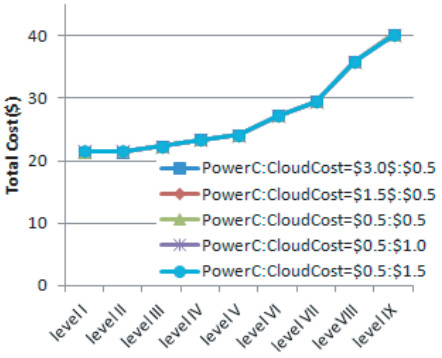

図 10　Total Cost (F =1/20)

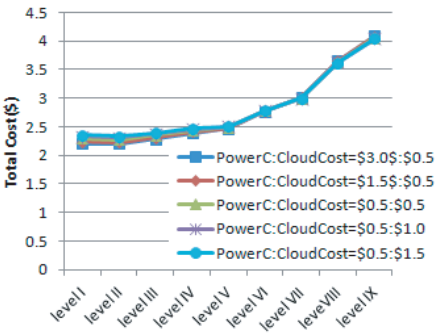

図 11　Total Cost (F = 1/200)

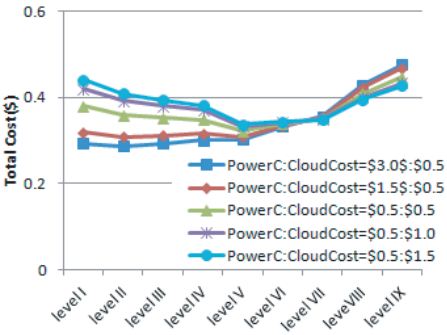

図 12　Total Cost (F = 1/2000)

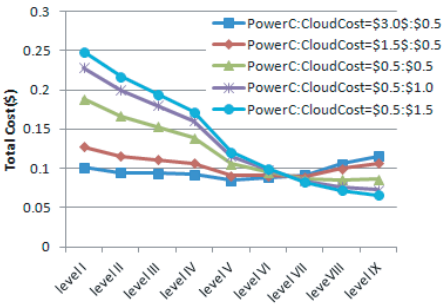

図 13　Total Cost (F = 1/20000)

7.　まとめと今後の課題

本研究では，ハイブリッドクラウド環境で，データインテンシブなジョブが次々と投入される場合において，それらのデータを効率よく処理しながら，さらにユーザによるパラメータ設定により，時間的コスト重視の負荷分散や，消費電力量を含む金銭的コストを抑える負荷分散を行えるミドルウェアを提案した．そして，そのミドルウェアを実際にパラメータを変化させて実行し，そのときのジョブの処理時間と，パブリッククラウドの従量制料金，プライベートクラウドの消費電力量料金を実際に測定，算出することで，それらを用いた時間的コストと金銭的コストを合わせたトータルコストの評価を行い，パラメータ設定により，実際にユーザが重要視するコストを抑える負荷分散を行えることを示した．また，時間的コストと金銭的コストの両方を抑えたい場合の負荷分散においても，本ミドルウェアで提供できることを示した．

今回の実験では L 値が固定されていたため，今後としては L

値や U 値のパラメータセットを様々に変化させた実験を行い，最適な負荷分散を考えていきたい．

8. 謝　　辞

本研究は一部，文部科学省科学研究費基盤研究「電力消費を制御するスケーラブルな情報の蓄積と検索」によるものである．また，本論文作成にあたり，独立行政法人 産業技術総合研究所 竹房 あつ子氏，中田 秀基氏，高野 了成氏，工藤 知宏氏には，大変有用なアドバイスをいただきました．深く感謝いたします．

文　　献

[1] Eucalyptus:http://www.eucalyptus.com/
[2] Xen:http://www.xen.org/
[3] KVM:http://www.linux-kvm.org/
[4] SHW3A:http://www.system-artware.co.jp/shw3a.html
[5] 豊島 詩織， 山口 実靖， 小口 正人：”データインテンシブアプリケーション実行時のクラウドリソースとローカルクラスタ間における負荷分散ミドルウェア”，日本データベース学会論文誌, Vol.10, No.1, pp.31-36, 2011 年 6 月
[6] 笠江 優美子， 豊島 詩織， 小口 正人：”Eucalyptus を用いたプライベートクラウドの様々な条件における消費電力量評価”，マルチメディア，分散，協調とモバイル (DICOMO2011) シンポジウム，3H-1， pp.550-557，天橋立宮津ロイヤルホテル，2011 年 7 月．
[7] G.JungK. R.Joshi, M.A.Hiltunen, R.D.Schlichting and C.Pu.:”Generating Adaptation policies for Multi-Tier Applications in Consolidated Server Environments”, In Proc. 5th IEEE International Conference on Autonomic Computing (ICAC2008), p23-32, June, 2008.
[8] E.Kalyvianaki,T.Charalambous, and S.Hand, ”Self-Adaptive and Self-Configured CPU Resource Provisioning for Virtualized Servers Using Kalman Filters”, In Proc. 6th International Conference on Autonomic Computing and Communications (ICAC2009)”, June, 2009
[9] C.; Parr, G.; McClean, S: .,”Energy-aware data centre management”, Communications (NCC), 2011 National Conference, pp.1-5, Jan, 2011
[10] Che-Yuan Tu, Wen-Chieh Kuo, Wei-Hua Teng, Yao-Tsung Wang, Shiau, S.; , ”A Power-Aware Cloud Architecture with Smart Metering”, Parallel Processing Workshops (ICPPW), 2010 39th International Conference, pp.497-503, Sep, 2010
[11] Mochizuki.K,Kuribayashi.S , ”Evaluation of Optimal Resource Allocation Method for Cloud Computing Environments with Limited Electric Power Capacity ”, Network-Based Information Systems (NBiS), 2011 14th International Conference, pp.1-5, Sep, 2011
[12] Gueyoung Jung; Hiltunen, M.A.; Joshi, K.R.; Schlichting, R.D.; Pu, C., ”Mistral: Dynamically Managing Power, Performance, and Adaptation Cost in Cloud Infrastructures ”, Distributed Computing Systems (ICDCS), 2010 IEEE 30th International Conference, pp.62-73, June, 2010
[13] Luna Mingyi Zhang; Keqin Li; Yan-Qing Zhang,”Green Task Scheduling Algorithms with Speeds Optimization on Heterogeneous Cloud Servers ”, Green Computing and Communications (GreenCom), 2010 IEEE/ACM Int’l Conference on and Int’l Conference on Cyber, Physical and Social Computing (CPSCom) , pp.76-80, Dec, 2010