サーバレスクラスド基盤ソフトウェアにおける ノードスケジューリング

2009SE150 萬田広大　2009SE270 鈴木祐典　2010SE130 守屋邑里
指示教員:宮澤 元

1　はじめに

近年，ネットワークを経由して様々なサービスを提供するクラウドコンピューティング (以下クラウド) を使用したコンピュータ利用が増えている．クラウドはクラウド基盤ソフトウェアを用いて複数のサーバコンピュータを集約することで実現されている．クラウドの利用モデルはパブリッククラウドとプライベートクラウドの 2 つに大別される [1]．パブリッククラウドは多種多様な組織や個人といった，一般利用者を対象として，インターネットを介して広い範囲でサービスを提供できる．一方，プライベートクラウドは学校や企業といった閉じた環境で使用され，同じ学校や企業，グループなどに対してサービスを提供するために構築されている．パブリッククラウドでは巨大なデータセンターを利用するなどの大規模な設備を用いて構築されるのでリソースの制約が比較的少ないのに対して，プライベートクラウドでは組織内のリソースのみを用いるので，利用できるリソースの制約がパブリッククラウドに比べて大きい．

プライベートクラウドのリソース不足を解消するためにはサーバとして利用できるリソースを追加する必要がある．しかし，サーバ用のコンピュータを追加するのはコストがかかるので，我々はクラウドのノードとして利用されていないクライアントのリソースに着目した．

我々はプライベートクラウド環境を想定して，組織内のリソースを統一的に管理してクラウドサービスを提供するサーバレスクラウド基盤ソフトウェアを開発している．このソフトウェアを用いることによって，各コンピュータをサーバやクライアントといった特定の用途のみに利用するのではなく，各コンピュータのリソースを必要な処理に適切に割り当てて利用できる．

本稿ではサーバレスクラウド基盤ソフトウェアにおけるノードスケジューリングについて述べる．ノードとなる各コンピュータをリソース量に応じてランク分けするとともに，仮想マシンサイズと仮想マシンの起動要求元に応じて適切なノードスケジューリングを行う．

2　従来のクラウド基盤ソフトウェア

クラウド基盤ソフトウェアはクラウドの処理基盤を提供するためのソフトウェアである．クラウド基盤ソフトウェアはサーバ仮想化技術を利用して，リソースを有効活用することにより，クラウドに対する様々な要求に柔軟に対応できる．

2.1　構成と動作

従来のクラウド基盤ソフトウェアでは，一般的に以下のようなコンポーネントを用いてクラウド環境が構成されている (図 1)．Cloud Controller(CLC) はクラウド全体の情報の管理を行い，ユーザに API や Web 管理画面を提供する．Cluster Controller(CC) は各ノードにおける仮想マシンの起動管理と仮想マシン同士の仮想ネットワークの管理を行う．Node Controller(NC) は各ノードで動作し，ノードにおける仮想マシンの動作を制御する．Storage Controller(SC) は仮想マシンに対してデータ保存用のストレージを提供する．USER は利用者が用いるクライアントコンピュータである．

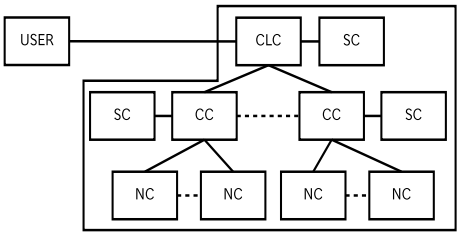

図 1　従来のクラウド基盤ソフトウェアの構成

仮想マシンの起動要求が来ると，クラウド基盤ソフトウェアは以下のように起動要求を処理する．

1. CLC はユーザから仮想マシンの起動要求を受け，CC にその要求を送信する．
2. CC は CLC から要求を受け取り，NC の空きリソース状況に応じて仮想マシンの起動，停止，削除の要求を送信する．これをノードスケジューリングと呼ぶ．
3. NC は CC から起動要求を受けハイパーバイザ上でインスタンスの起動，停止，削除を行う．
4. 起動した仮想マシンは CC で起動している DHCP サーバから IP アドレスを取得する．

実際に利用されているクラウド基盤ソフトウェアには様々なものがある．Amazon 社による AmazonEC2[2] や AmazonS3[3] といった有料のサービスに加え，Eucalyptus[4] や Cloud Stack[5]，Open Stack[6] といったオープンソースのプロジェクトが存在している．

2.2　ノードスケジューリング

CC は CLC からの仮想マシン起動要求に応じてノードスケジューリングを行って NC に仮想マシンを起動させる．NC を選択するポリシーは様々なものが考えられる

が，ここでは代表的な 2 つを示す．

GREEDY ポリシーは，1 つの NC に対してリソースが枯渇するまでその NC で仮想マシンを起動し，リソースが枯渇すると次の NC を使用する．図 2 で，VM の番号が GREEDY ポリシーにおける仮想マシンの起動順となる．

ROUNDROBIN ポリシーは，NC を順番に選んで仮想マシンを起動し，最後の NC の仮想マシンを起動すると，最初の NC をもう一度使用する．空きリソースが枯渇すると次の NC を使用する．図 3 で，VM の番号が ROUNDROBUIN ポリシーにおける仮想マシンの起動順となる．

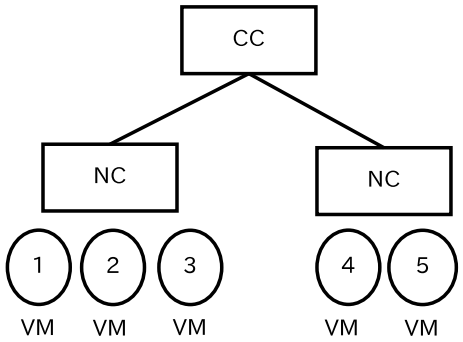

図 2 GREEDY ポリシーによるノードスケジューリング

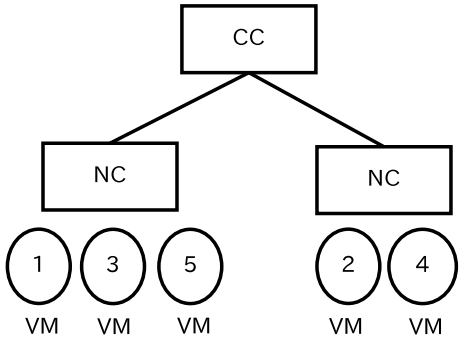

図 3 ROUNDROBIN ポリシーによるノードスケジューリング

2.3 クライアントコンピュータをノードとする際の問題点

組織内で閉じたプライベートクラウド環境では，利用できるリソースが不足する可能性がある．この問題はリソースを追加することで解決できるが，そのためには時間とコストがかかる．そこで，既にあるクライアントの空きリソースをクラウドで利用するとリソース不足が緩和できると期待できる．しかし，従来のクラウド基盤ソフトウェアのノードスケジューリングで単純にクライアントのリソースを利用しようとすると，ユーザインターフェース処理などの非クラウド処理を考慮しないので，クライアントの空きリソースをすべて使いきってしまう恐れがある．また，ノードの差異を考慮しないので，適切なスケジューリングが行われない可能性がある．

3 サーバレスクラウド基盤ソフトウェアにおけるノードスケジューリングの設計

2.3 節の問題を解決するために我々はサーバレスクラウド基盤ソフトウェアを開発している．プライベートクラウド環境において効率的にコンピュータリソースを利用するためにクライアントコンピュータをノードとして利用する．そこで，クライアントコンピュータごとのリソースの違いを考慮して適切なノードで仮想マシンを起動するスケジューリングを行う．

3.1 サーバレスクラウド基盤ソフトウェアの構成

我々が開発するサーバレスクラウド基盤ソフトウェアは従来のクラウド基盤ソフトウェアと異なり利用するノードにクライアントコンピュータが含まれる．

図 4 のようにクライアントコンピュータをサーバコンピュータと同等のノードとして使用し，NC を動作させるのでクライアントコンピュータのリソースを使用することが可能である．

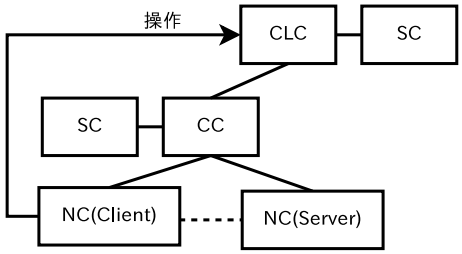

図 4 サーバレスクラウド基盤ソフトウェアの構成

3.2 ノードスケジューリング

サーバレスクラウド基盤ソフトウェアではクライアントコンピュータとサーバコンピュータを区別なく使用するが，両者のコンピュータリソースには違いがあり，これを考慮してノードスケジューリングを行う必要がある．

3.2.1 ノードのランク分け機能

一般にプライベートクラウドではノードのリソースが均一ではないので，各ノードのリソースを比較して仮想マシンをいくつ起動できるかを元にしたランク分けを行う．これにより，リソースが潤沢に存在するノードコントローラで仮想マシンを起動するようなノードスケジューリングを行う．すなわち，ノードスケジューリングに際してリソースが潤沢なより上位のランクに存在するノードを優先的に使用する．また，リソースが不足して仮想マシンが起動することができないノードをあらかじめノードスケジューリングの対象から除外することもできる．

3.2.2 ノード予約機能

仮想マシンを起動したいユーザが使用しているクライアントコンピュータ上で仮想マシンを起動するためにノード予約機能を実装する．これは，ノードスケジューリングに

際してあらかじめ予約したノードを優先的に利用する機能である．仮想マシンを起動したいユーザは使用しているクライアントコンピュータの情報をノード予約機能を用いて登録する．そして，ノードスケジューリングの際に優先的に予約されたノード上で仮想マシンを起動することにより，仮想マシン起動による負荷を他のユーザが利用しているクライアントコンピュータにかけることを避ける．

3.3 サーバレスクラウド基盤ソフトウェアにおける仮想マシン起動処理

仮想マシンを起動する方法は二つ存在する．ノード予約機能で予約したノード上で仮想マシンを起動する場合とノード予約機能で予約したノードが存在しない，もしくはノードの空きリソースが存在しないという理由で他のノードで起動する場合である．

前者の場合を図 5 に示す．仮想マシン VM1 は起動要求したクライアントコンピュータに空きリソースが存在するのでそのクライアントコンピュータ上で起動する．

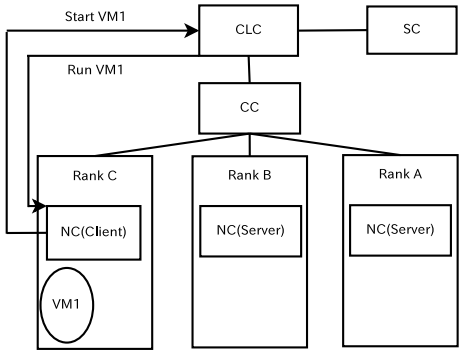

図 5 予約したノードにリソースが存在する場合

後者の場合の動作を図 6 に示す．予約したクライアントコンピュータ上に空きリソースが不足しているので仮想マシン VM3 はランク分けされたノードコントローラの内上位にランク分けされたノードコントローラで起動する．

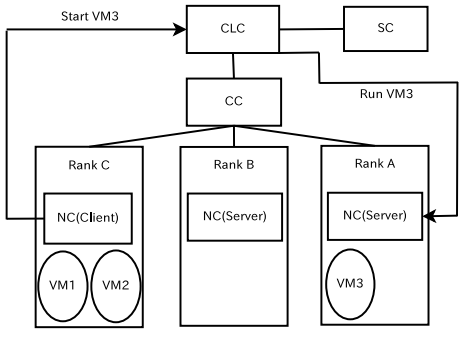

図 6 予約したノードにリソースが存在しない場合

4 ノードスケジューリングの実装

我々は本システムの実装を Eucalyptus3.2 をベースにして行った．

4.1 SERVERLESS スケジューリングポリシーの実装

我々が述べたスケジューリング機構を SERVERLESS スケジューリングポリシーとして実装を行う．このスケジューリングポリシーは先に述べたノードのランク分け機能とクライアント予約機能を実装したものである．

4.2 ノードのランク分け機能

各ノードをメモリー量，HDD 容量，コア数を元に仮想マシンのレベルや個数でランク分けを行う．今回の実装では表 1 のように仮想マシンの起動可能個数でランク分けを行う．Eucalyptus で使用する仮想マシンにはランクがあり最小の仮想マシンはメモリ量 512MB，HDD 容量 5GB，コア数 1 コアであるので Rank A は空きリソースがメモリ量 4GB 以上、HDD 容量 40GB，コア数 8 コアになりリソースが潤沢にあるといえる．

ランク	仮想マシン起動個数
Rank A	8 個以上
Rank B	4 個以上
Rank C	1 個以上
NoUse	0 個

表 1 ランク分け基準表

ランクが上位のノードから仮想マシンの起動に使用される．同じランク内のノードの性能は同等だと考えられるのでリソース使用率の偏りを少なくするためにラウンドロビンで仮想マシンを起動する．

4.3 ノード予約機能

今回の実装ではノード予約機能をファイルを用いて実装した．ノードの登録ファイルに書き込まれたノードの IP アドレスを使用してノードを選択する．今回の実装はファイルを用いて行ったのでファイルの重複書き込みや書き込みミスによる人為的ミスを避けなければならない．また，仮想マシンが起動するまではファイルの削除がされてはならず仮想マシンが起動した場合はファイルが削除されなければならない．

5 実験

本システムの有効性を確認するため従来のポリシーと比較による実験を行った．

5.1 スケジューリングポリシーによるノード選択時間の比較

ノードを 1 台のみ用いた環境で既存の GREEDY ポリシー，ROUNDROBIN ポリシーと我々が実装した SERVERLESS ポリシーでノード選択にかかる時間を計測した (図 7)．GREEDY ポリシー，ROUNDROBIN ポリシー，SERVERLESS ポリシーの順にノード選択にかかる時間が増えていく．これは各ポリシーの実装におけるファ

イル操作の回数を反映したものである．SERVERLESS ポリシーではノード予約機能の影響で頻繁にファイル操作が行われているので時間がかかる．

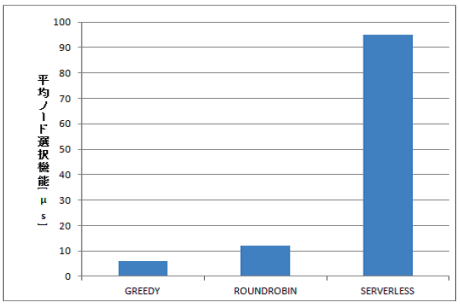

図 7　ポリシーによる平均ノード選択時間の比較

5.2　ノード数によるノード選択時間の比較

ノードが複数存在する環境で既存の GREEDY ポリシー，ROUNDROBIN ポリシーと我々が実装した SERVERLESS ポリシーにおけるノードの数によるノード選択時間の変化を示したものが図 8 である．SERVERLESS ポリシーではノードの数が増えることによりファイル操作が増えるのでノード選択に時間がかかる．GREEDY ポリシーと ROUNDROBIN ポリシーはノード数が増えることによる時間の変化は見られず図 7 と同じ結果となった．一方，SERVERLESS ポリシーでは適切なノードに仮想マシンが作成されることが確認できる．

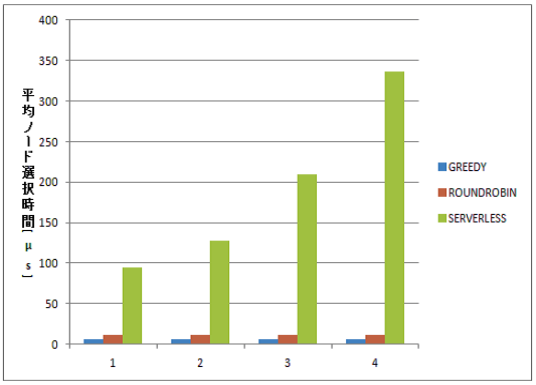

図 8　ノード数による平均ノード選択時間の比較

5.3　考察

我々が提案する SERVERLESS ポリシーではノードをランク分けに従って使用し，適切なノードに仮想マシンを作成することができる．しかし，ROUNDROBIN ポリシーのファイル操作 1 回に比べて SERVERLESS ポリシーはファイル操作が 6 回以上行われるのでノード選択時間が増えている．これはファイルを使用せずに実装を行うことで解決できると考えられる．

6　関連研究

OS のスケジューリングにおいてラウンドロビンスケジューリングは，プロセスに優先度がないという前提で使用されるが，プロセスに優先度がある場合，優先度付きラウンドロビン [7] と呼ばれる優先度に応じたラウンドロビンスケジューリングを使用する場合がある．我々のノードスケジューリングにおけるノードのランク分けは，クラウドのスケジューリングに優先順位付けラウンドロビンを適用したものと考えることもできる．

7　まとめと今後の課題

プライベートクラウドのクライアントコンピュータをサーバコンピュータと同等のノードとして利用するためのノードスケジューリングを提案した．ノードのランク分け機能とノード予約機能を用いることで効率的なコンピュータリソースの利用を可能とした．Eucalyptus に新しいスケジューリングポリシーを追加する形で実装を行い，実験により正しく動作することを確認した．一方，実装の問題でノード選択の速度は遅い．

今後の課題は実装を改良しノードスケジューリングの速度を向上をするとともに，クラウド全体を対象としたノード予約機能を実装する．

参考文献

[1] 加藤英雄:『クラウドコンピューティング-サーバは雲のかなた』．共立出版，東京，2011.

[2] Amazon EC2, “Amazon EC2”, http://aws.amazon.com/EC2

[3] Amazon S3, “Amazon S3”, http://aws.amazon.com/S3

[4] Daniel Nurmi, Rich Wolski, Chris Grzegorczyk Graziano Obertelli, Sunli Soman, Lamia Youseff, Dmitrii Zagorodnov;”The Eucalyptus Open-source Cloud-computing System”. In *Proceedings of the 9th IEEE/ACM International Symposium on Cluster Computing and the Grid(CCGRID’09)*

[5] CloudStack, “Apache CloudStack”, http://cloudstack.apache.org/

[6] OpenStack, “openstack CLOUD SOFTWARE”, http://www.openstack.org/

[7] A・S・タネンバウム (水野 忠則・太田 剛・最所 圭三・福田 晃・吉澤 康文 訳) ,『モダンオペレーティングシステム』, ピアソン・エデュケーション，東京，2004.