Fog コンピューティング基盤のためのストレージシステムの提案

2013SE024 蜂谷悠海　2013SE100 倉知見多
指導教員：宮澤元

1　はじめに

クラウドコンピューティング (クラウド) が登場して約 10 年経ち，今ではクラウドが欠かせない時代になっている．クラウドでは，従来手元のコンピュータで管理・利用していたようなソフトウェアやデータなどを，インターネットを通じてサービスの形で提供する．利用者は必要なサービスを必要なときに必要なだけ利用することができる．また，クラウドストレージでは，複数のストレージサーバを活用して，大量のデータを高い信頼性で高速に扱うことが出来る．そして，センサーなどとクラウドを接続させ，センサーなどでコンピュータなどの情報・通信機器だけでなく，世の中に存在する様々な物体（モノ）から得たデータをクラウドで分析，処理をさせてヒトや機械にフィードバックする Internet of Things（IoT）という技術にも活用されている．

このように近年では，利用者が大幅に増加したり，アプリケーションの性質が多様化したりすることにより，クラウドへの負荷が増加しており，ネットワーク帯域の制約やネットワーク遅延（レイテンシ）が問題とされている [3]．特にレイテンシの問題の場合，ファイル転送やビデオ再生，Web 検索等ある程度のレスポンスの遅れが許容される場合にはクラウドでも処理は可能であるが，よりレイテンシの制約が厳しいアプリケーションの要求をクラウドで処理すると，許容された時間内では処理しきれない場合がある．

そこでフォグコンピューティングと呼ばれる分散処理環境をクラウドとデバイスの間のデバイスに近い所に設置することが提案されている [1]．デバイスの要求をデバイスの近くで処理することによって，レイテンシの問題を解決するとともに，IoT のデータ前処理をフォグで行うことによって，クラウドに送信するデータ量の削減が可能となる．このような処理を行うために，フォグはデータの処理機能だけではなく，データを保持するためのストレージを備える必要がある．

しかし，クラウドストレージシステムをこのようなフォグ上のストレージにそのまま適用すると問題を生じる場合がある．従来のクラウドストレージシステムは，ネットワークレイテンシが均一な環境で使われることを想定しているので，データを保存するサーバを選択する際にネットワークレイテンシを考慮しない．もし，データを保存するサーバのレイテンシが大きかった場合，ファイルアクセスに伴って発生するレイテンシが問題となる．そこで，低レイテンシが求められるフォグのストレージシステムでは，データを保存するサーバを選択する際に，ネットワークレイテンシを考慮する必要がある．

我々は，フォグコンピューティング環境において複製保存する場所を工夫することにより，低レイテンシでのアクセスを保証するクラウドストレージのアーキテクチャを提案する．クラウドストレージがデータの複製をとる構造になっていることを利用することでフォグとクラウドの両方のストレージサーバに複製を保存して，低レイテンシでのアクセスを可能とするとともに，冗長性も確保できる．我々はフォグコンピューティングの環境下でのクラウドストレージのエミュレータのプログラムを作成する．エミュレータを用いた実験により，提案システムの有効性を示す．

2　研究の背景

ここでは，クラウドストレージシステムに用いられる分散ストレージ技術とフォグコンピューティングについて述べる．

2.1　分散ストレージ技術

クラウドでは，複数のストレージに分散してデータを格納する分散ストレージ技術が用いられる．分散ストレージ技術を用いると，ストレージアクセスを並列に行うことにより，アクセス性能を向上することができる．また，複数のストレージを利用して冗長性を確保することによって，耐障害性も向上できる．本節では，分散ストレージ技術を応用した代表的なシステムとして，HDFS について述べる．

2.1.1　HDFS

HDFS は，Hadoop が利用している分散ファイルシステムである．ファイルをブロック単位に分割（初期設定では 64MB）して複数のデータノードと呼ばれるデータの読み込みや書き込み処理を行うサーバに保存する．そうすることで，ファイルの読み書きを高速化することが可能となる．また，それらのデータノードを管理するのがネームノードと呼ばれるサーバで，クライアントからの指示に従ってデータノードに指示を送る．読み込み時はクライアントとデータノードが直接通信をする．ブロックの複製を複数の場所に同時に保存（初期設定では 3 つ）することで耐障害性に優れるが，3 倍の記憶容量を消費してしまう．

2.2　フォグコンピューティング

近年，IoT が普及し始めており，2020 年には 500 億台のデバイスがクラウドにつながるとされている [2]．500 億台のデバイスから得られるデータ量は膨大で，そのデータがクラウドに集中してしまうと，データ処理が追いつかなくなってしまう．そこで，シスコシステムズ社によって提唱されたのが，フォグコンピューティング（フォグ）である．クラウドとデバイスの間にフォグと呼ばれる分散処理

環境を置くことにより，クラウドにデータを送る前に，デバイスに近いところで大量のデータを処理し，データ量を減らしてからクラウドに送ることによりクラウドの負担を減らしている [1]．クラウド（雲）よりもデバイスに近いのでフォグ（霧）と呼ばれている．

3　フォグコンピューティングを考慮したストレージシステム

我々はフォグコンピューティングを考慮したストレージシステムを提案する．1 節で述べた問題を解決するために，VM がデータを保存する際に以下の三つに複製を保存するようにする．

- 現在クライアントが使用しているフォグ上のストレージサーバ
- 現在クライアントが使用しているフォグ内の別のストレージサーバ
- クラウド上のストレージサーバ

これによって，耐障害性を高めつつ，低レイテンシを保ち，保存や読み出しをすることがが可能である．図 1 にファイルを現在使用しているフォグのストレージに 2 つ，クラウドのストレージに 1 つ保存した場合を示す．現在使用しているフォグのストレージサーバにファイルが保存されるため，保存，読み出しをする際に遅延を気にすること無く利用することができる．

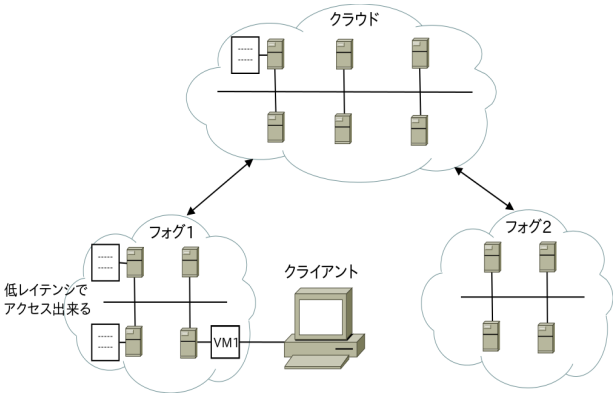

図 1　フォグにファイルがある場合

また，提案アーキテクチャではサーバからのレスポンスのタイミングも低レイテンシでのデータアクセスを行うために再検討する．通常のクラウドストレージと同様に，三箇所のサーバ全てに保存が完了した後，レスポンスを返してしまうと，レイテンシが高くなってしまうことが問題になる．そこで，クライアントが現在使用しているフォグのストレージに保存が完了したらクライアントにレスポンスを返し，他二つの保存処理をバックグラウンドで行う．これにより，クライアントに対してレイテンシを損なうことがなく通信することが可能になる．しかし，クライアントにバックグラウンドで保存する二つのデータの安全が保証されないので現在使用しているフォグから送るデータがクラッシュしてしまった場合も問題になる．この問題を改善するために，2 つ目以降のサーバに現在使用しているフォグのファイルサーバを選ぶ．データ保存を低レイテンシで行うことが出来るので，クラッシュによってデータを喪失する危険性を低減できる．

3.1　保存

ファイルを保存する場合，まずクライアントとフォグのストレージサーバを接続する．次にクライアントからファイルを固定サイズで分割して送る．分割したファイルの保存が完了したら，ストレージサーバはメタデータサーバ（MDS）に 1 台目のサーバの IP アドレスを送信し，クライアントにレスポンスを返す．この時点でクライアントとフォグのストレージサーバは通信を切り，最初に通信したフォグのストレージサーバが，バックグラウンドで現在使用しているフォグ内の別のファイルサーバやクラウドのストレージサーバに固定サイズで分割したファイルを送信し，保存する．二つ分保存し終わり，レスポンスが返ってきたら，MDS に保存したファイル名を送り，保存したサーバの IP アドレスをメタデータとして保存する．図 2 にファイルを保存する際の処理の流れを示す．クライアントである VM は最初にフォグ 1 というストレージサーバに接続してファイルを保存する．フォグ 1 は MDS にメタデータを保存して VM にレスポンスを返した後，フォグ 2 とクラウドにファイルの複製を保存する．

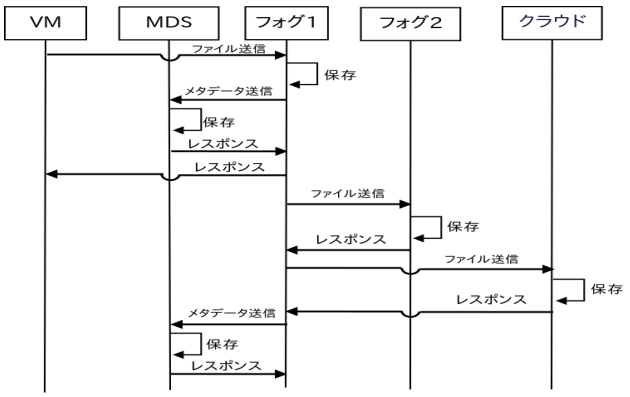

図 2　保存時の流れ

3.2　読み出し

ファイルを読み出す場合，クライアントは MDS に読み出したいファイル名を送る．MDS は送られてきたファイル名をもとにメタデータが存在するかを確認する．メタデータが存在すれば，分割されたファイルを持つストレージサーバのリストを取得する．ストレージサーバのリストをクライアントに送り，クライアントはその中からネットワーク距離が一番近いストレージサーバを選び接続し，ファイルを受け取る．もしファイルがサーバのクラッシュ等で受け取れない場合は次にネットワーク距離が近いファイルサーバに接続し，ファイルを受け取る．また，そのス

トレージサーバのファイルもクラッシュ等で受け取れない場合も同様に処理を行う．

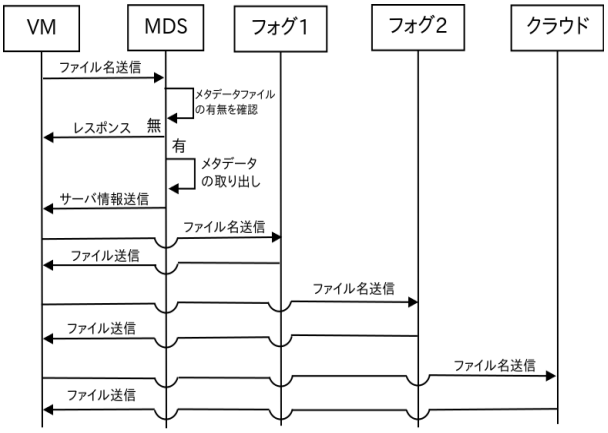

図 3　読み出し時の流れ

4　実装

実用システムを実装する前段階として，C 言語で socket を用いたクライアントサーバ方式のエミュレータを作成した．サーバは IP アドレスを使用し，固有の IP アドレスで判別して個別に通信を行う．現在ファイルの保存と読み出しの二つの動作をエミュレート出来る．

ただし，ファイルの保存については，1 台目のサーバに保存完了後，MDS へのメタデータの更新をする処理は実装していない．また，ファイルの読み出しについては，実装の簡単化のために，ファイル転送を MDS が中継する実装となっている他，障害発生時にストレージサーバを再選択する機能も実装していない．

5　実験

提案したストレージシステムのアーキテクチャの有効性を示すために実装したエミュレータを用いた実権を行った．クライアントがサーバに対してさまざまなサイズのファイルを読み書きする際の実行時間を計測した．実験で変化させる条件を表 1 に示す．

表 1　実験条件

ファイルサイズ	1GB	1MB	1KB	
サーバの種類	全部フォグ	クラウド 1 台	クラウド 2 台	全部クラウド
ネットワーク遅延	1ms	5ms	10ms	
レスポンス方法	1 つ目のサーバ	3 つ目のサーバ		

ファイルサイズ，サーバの種類，ネットワーク遅延，レスポンス方法の条件を任意に組み合わせて実験を行う．時間の計測は time コマンドを用いて行い，実行時間として real の値を 10 回測定し，平均を取った．ネットワーク遅延は tc コマンドを使用して設定し，遅延の値はフォグのサーバを想定した 0ms と，クラウドを想定した 1ms, 5ms, 10ms を用いた．

5.1　実験環境

実験で使用した計算機の仕様を表 2 に示す．VM と見立てたクライアントを 1 台とサーバを 4 台使用した．サーバの 1 台を MDS とし，残りのサーバをストレージサーバに用いる，

表 2　実験に使用した計算機の仕様

	クライアント	サーバ (1,2,3)	MDS
CPU	core(TM) i5-3320	core(TM) i7-3770	
OS	Ubuntu 12.04	Ubuntu 16.04	Ubuntu 14.04
HDD	320GB	NFS サーバ	
NIC	ギガビットイーサネット		
メモリ	4GB	8GB	
コア数	2 コア	4 コア	
クロック周波数	2.6GHz	3.4GHz	

5.2　ファイル保存実験

ファイルの保存実験の結果を図 4，5，6 に示す．グラフの横軸はサーバの種類を示し，縦軸は実行時間を示している．

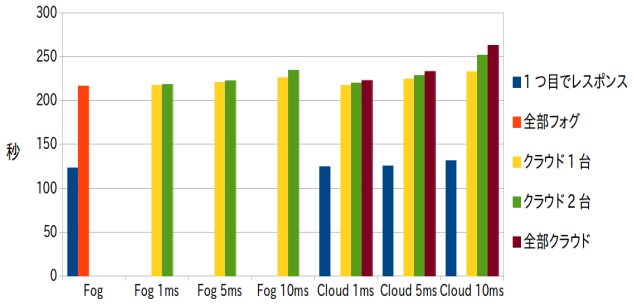

図 4　1GB のファイルを分散保存したときの実行時間

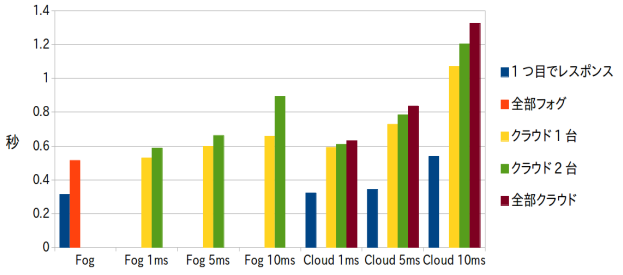

図 5　1MB のファイルを分散保存したときの実行時間

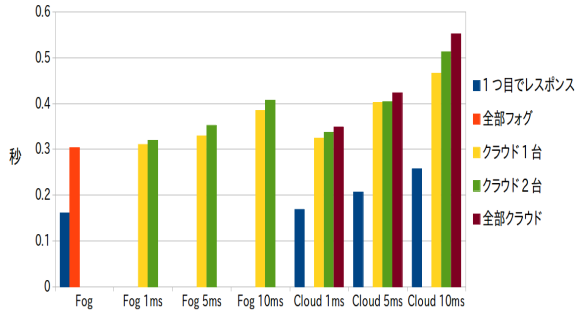

図 6　1KB のファイルを分散保存したときの実行時間

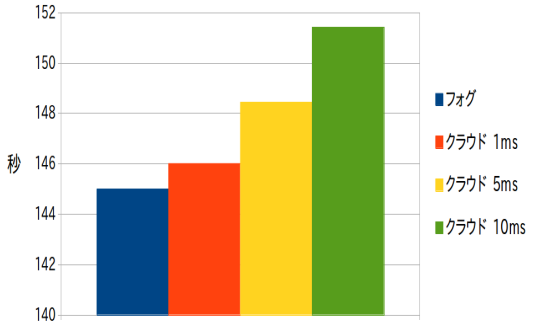

図 7　1GB のファイルの読み出しをしたときの実行時間

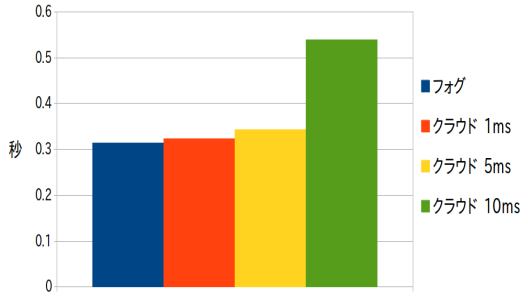

図 8　1MB のファイルの読み出しをしたときの実行時間

ファイルを保存する場合，ファイルサイズに関わらずネットワーク遅延が大きいほど実行時間が長くなる結果になった．

ファイルサイズに対する実行時間の関係を割合で見ると，1M バイトのファイルを保存するときが，一番実行時間の上げ幅が大きく，次いで 1K バイト，1G バイトと続く結果になった．1G バイトの場合の実行時間の上げ幅が小さい理由は，最初のクライアントからサーバに接続する時間が一番長く，その後のサーバ間のファイルの保存処理にはあまり時間がかからないので，その時間がファイルの大きさが大きくなるほど長くなるからである．また，1M バイトの場合が 1K バイトの場合よりも実行時間の上げ幅が大きい理由は，1K バイトのファイル送信の際にかかる時間が 1M バイトの場合に比べ短過ぎたからである．従って 1M バイト程度のファイルを保存する場合に一番遅延の影響が出ることが分かる．

ファイルの複製を保存する場所の観点から見ると，サーバのネットワーク遅延を大きく設定するほど顕著に実行時間の差が大きくなることが分かった．

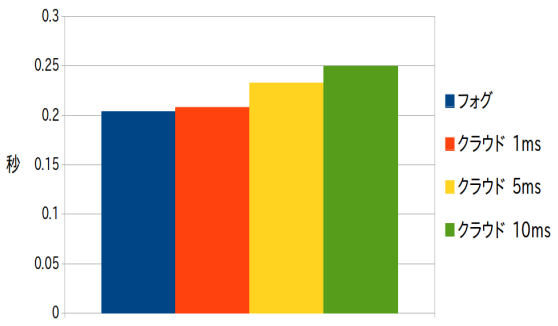

図 9　1KB のファイルの読み出しをしたときの実行時間

クライアントにレスポンスを返すタイミングの観点から見ると，三つファイルを保存し終わってからクライアントにレスポンスを返すよりも，一つ目のファイル保存終了時にクライアントにレスポンスを返す方が二倍近く実行時間が速くなることが分かった．この効果は遅延時間が伸びるほど差が出るので効果的であるといえる．

5.3　ファイル読み出し実験

ファイル読み出し実験の結果を図 7，8，9 に示す．図の横軸はサーバの種類を示し，縦軸は実行時間を示す． ファイルサイズに対する実行時間の関係を見ると，読み出し時については 1G バイト，1M バイト，1K バイトの順に実行時間の上げ幅の割合が大きい．よってファイルサイズが大きいほど遅延の影響は小さくなることが分かった．

6　まとめ

我々は，フォグコンピューティング環境においても低レイテンシでのファイル処理を実現できるクラウドストレージシステムのアーキテクチャを提案した．

フォグ環境下でのクラウドストレージシステムのエミュレータのプログラムを作成し，実験することで低レイテンシが実現出来ていることを確認した．

今後の課題は，作成したプログラムをシステムとして実装し，実用的なアプリケーションに対して有効に働くことを確認することである．

参考文献

[1] Flavio Bonomi, Rodolfo Milito, Jing Zhu, Sateesh Addepalli,“Fog Computing and Its Role in the Internet of Things”,in *Proceedings of the first edition of the MCC workshop on Mobile cloud computing*, PP. 13-16,2012

[2] `http://itnp.net/article/2014/03/03/629.html`(2016 年 9 月 28 日閲覧)

[3] `http://businessnetwork.jp/Portals/0/SP2016/cisco_fog/`(2016 年 9 月 28 日閲覧)