4L-07

オブジェクトストレージにおけるオブジェクト数と性能に関する一考察

早川峻平†1　　山口実靖†1
工学院大学 工学部 情報通信工学科†1

1 はじめに

今日ではストレージに保存するデータ量の増加とデータの性質の変化に伴い，そのデータを保存する方法が重要となっている．ブロックストレージは電子メールや動画像といった非構造化データの保存には不向きであり，ファイルストレージのディレクトリ構造ではデータ量の増加によるデータの維持，管理が難しくなってきている．この問題を解決するためにデータをファイル単位やブロック単位ではなくオブジェクトという単位で扱うオブジェクトストレージが提案された．オブジェクトストレージはディレクトリ構造を持たず，オブジェクトには URI が付与されデータとそれを扱うメタデータで構成される．オブジェクトストレージでは，ファイルストレージにおけるディレクトリと類似したものであるコンテナを作成し，そのコンテナに対し HTTP/REST API を用いてファイルをアップロード，またはダウンロードを行う．本論文では，OpenStack Swift [1]を用いたオブジェクトストレージにおけるオブジェクト数と性能についての考察を行う．

2 ネットワークストレージ

データのバックアップなどのストレージの管理は企業などの組織にとって重要な事項であるが，計算機に直接接続されたストレージ(DAS, Direct Attached Storage)を用いている場合はそれぞれを個別に管理するのに膨大なコストがかかるという問題がある．この問題を解決するために，NAS (Network Attached Storage) や SAN (Storage Area Network)が提案され，ストレージが一元化され集中的に管理できる様になった[2]. NAS はファイルレベルのストレージアクセスを，SAN はブロックレベルのストレージアクセスを提供する．しかし，これらを用いてもデータはディレクトリ構造を持つファイルシステムで管理され，膨大な数のデータを管理することは困難であり，さらなる改善が期待される様になった．

A Study on Performance of Number of Objects with Object Storage
†1. Shunpei Hayakawa, Saneyasu Yamaguchi
†1. Department of Information and Communications Engineering, Kogakuin University

3 OpenStack Swift

OpenStack Swift はクラウド環境を構築するためのソフトウェア開発プロジェクトである OpenStack のコンポーネントの一つである．Swift はプロキシノード，アカウントノード，コンテナノード，オブジェクトノードの複数のノードによって構成され，各ノードはそれぞれストレージにアクセスするための API の提供や各サービスの管理，アカウント情報の管理，コンテナ情報の管理，オブジェクトの管理の機能を提供している．オブジェクトの操作を行う場合，まずクライアントからの要求をプロキシノードが受け，プロキシノードは他ノードと連携を行いクライアントからの要求を処理する．アカウントノードとコンテナノードは情報を管理するために SQLite3 データベースを用いており，アカウントノードはコンテナ名の一覧，アカウント全体の統計情報などを管理し，コンテナノードはオブジェクト名の一覧，ACL 情報などの管理を行っている．また，オブジェクトストレージは，アップロードされたオブジェクトのレプリカを作成し分散して保存することにより，保存されたデータの安全性を保障する機能を有している．

4 Container Sharding

前章で述べた様に，OpenStack Swift ではコンテナにアップロードされた全てのオブジェクトの名前の一覧を SQLite3 データベースに格納している．コンテナ内のオブジェクト数が増加すると，データベースファイルのサイズも大きくなる．これにより，すでに大量のオブジェクト名が格納された SQLite3 ファイルに新たにオブジェクト名を格納する際に，このファイルの読み込みのための待ち時間が増加する．Container Sharding はコンテナに一定数以上のオブジェクトがアップロードされた場合，そのコンテナを分割し，上記の問題を解決する手法である．

5 性能評価

本章にて，Swift におけるオブジェクト数とコンテナ数とオブジェクト追加の関係の調査を行

う．図 1 に実験環境，表 1 に実験環境の詳細を示す．Proxy Node，Object Node，Client の 3 台の物理マシンと，Object Node 上に OpenStack のサービスを使用する際のユーザ認証を担う Controller Node を稼働させた VM でオブジェクトストレージシステムを構成した．本実験では，Object Node 上にアカウントサービス，コンテナサービス，オブジェクトサービスを実行できるように構築し，レプリカの作成は行わないように設定した．

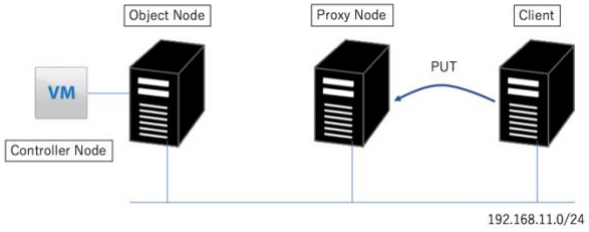

図 1 実験環境

表 1 実験環境の詳細

	Object Node	Proxy Node	Client	Controller Node (VM)
OS	Ubuntu 16.04			
CPU	Intel Core i5-4670 3.40GHz	Intel Celeron 2.27GHz×2	AMD Athlon(tm) II X2 220 Processor	2
メモリ	8GB	12GB	2GB	4GB
HDD	3TB	250GB	250GB	20GB
OpenStack Version	Pike			

Swift のベンチマークを行うツールである swift-bench を用いて，Client から PUT リクエストを送信した．オブジェクトサイズは 1KB とし，1 個のコンテナに合計 20 万オブジェクトをアップロードした場合と，200 個のコンテナを用意し各コンテナに 1000 オブジェクトずつ合計 20 万個のオブジェクトのアップロードを行った場合の 2 種類の測定を行った．この時，1000 オブジェクトをアップロードする毎に処理性能を測定した．

図 2 に 1 個のコンテナに合計 20 万オブジェクトをアップロードした場合と，200 個の各コンテナに 1000 オブジェクトずつアップロードを行った場合のリクエスト処理性能を示す．横軸がアップロードした合計オブジェクト数で，縦軸が処理性能である．10 万オブジェクト以下では毎秒 3.5 リクエストと高い処理性能を実現しているが，10 万オブジェクトを越えると 1 個のコンテナにアップロードした場合は性能が不安定となり，16 万オブジェクトで毎秒 1 リクエスト以下と性能が悪化していることがわかる．一方，コンテナを分割した場合では 10 万オブジェクト以上でも毎秒 3.5 リクエストと高い処理性能を維持できていることがわかる．

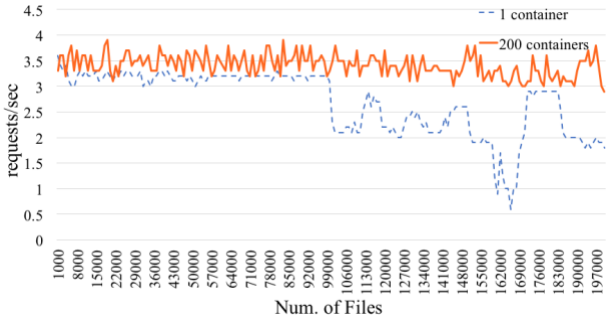

図 2 コンテナ数とオブジェクト数の requests/sec の関係

6 考察

前章の評価より，1 個のコンテナにオブジェクトをアップロードする場合，オブジェクト数が多い環境では性能が低下し不安定になった一方で，コンテナを分割した場合では常に高い性能を維持できることがわかった．コンテナを分割することにより，1 個のコンテナあたりのオブジェクト数が少なくなり，データベースファイルのサイズも小さくなる．その結果，SQLite3 ファイルに新たにオブジェクト名を格納する際にかかるファイルの読み込みの待ち時間が減少し，コンテナを分割した場合は処理性能が維持できたと考えられる．すなわち，Container Sharding の効果が得られたと考えられる．更なる性能向上には，適切なコンテナ数の調査などが有効であると考えられる．

7 おわりに

本論文では，OpenStack Swift を用いて環境を構築し，処理性能の評価を行った．性能評価により，コンテナを分割した場合においては 1 個のコンテナにアップロードをする場合よりも高い性能を維持することができることを示した．今後は，コンテナの分割や SQLite3 の保存方法を変更して性能を評価し，性能向上を図る予定である．

謝辞

本研究は，JST，CREST JPMJCR1503 の支援を受けたものである．

本研究は JSPS 科研費 25280022，26730040，15H02696 の助成を受けたものである．

参考文献

[1] Sridevi Bonthu, Y S S R Murthy, M. Srilakshmi “Building an Object Cloud Storage Service System using OpenStack Swift”, International Journal of Computer Applications (IJCA’14), 2014.

[2] S. Yamaguchi, M. Oguchi and M. Kitsuregawa, "Trace system of iSCSI storage access," *The 2005 Symposium on Applications and the Internet*, 2005, pp. 392-398. doi: 10.1109/SAINT.2005.65