1N-04

KVSにおける動的ノード追加時間の短縮に関する一考察

御代川 翔平† 徳田 大輝† 山口 実靖†

工学院大学大学院 工学研究科 電気・電子工学専攻†

1. はじめに

大規模SNSにかかる負荷は社会の動きに合わせて変動している．Facebookにおける時間単位ごとの投稿数は最多が18:00から19:00の間で10.53%を占め，また03:00から04:00の間が最少であり0.1%を占めている．そのため約100倍の投稿数差が生じている[1]．これら大規模SNSの普及などにより，データベース(DB)管理システムのスケーラビリティの確保や動的スケールが重要視されるようになり，分散型DBのKVS(Key-Value Store)が注目されるようになった．KVSはシンプルなデータベース構造であるため高いスケーラビリティを持つ．またサーバ稼働中に新規ノードの追加や既存ノードの削除が可能といった動的性能伸縮性を持っているため，高負荷時はノード数を増やしデータベース管理システムの性能を向上させ，低負荷時はノード数を減らし省電力化させるといったことができる．

本稿ではKVSの一つであるCassandraを用い，I/Oスケジューラや KVS実装(バージョン)と，ノード追加時間の関係について考察する．

2. Key-Value Store

KVS(Key-Value Store)は，Keyを指定してValueを得る仕組みのデータベース管理ソフトウェアである．機能がRDBMSなどより単純になっているが，高いスケーラビリティを得ることができる．KVSの代表的な実装の一つにCassandra[2]がある．

Cassandra は Facebook 社が開発した KVS であり，現在は Apache Software Foundation のトップレベルプロジェクトである．

Cassandra のノード配置には仮想ノードを用いる手法と，用いない手法の二通りの方法がある．本稿では，前者を仮想ノード配置と，後者を非仮想ノード配置と呼ぶ．まず，非仮想ノード配置について述べる．Cassandra を構成する各ノードはトークンと呼ばれるハッシュ値を持ち，リング状のハッシュ空間にトークンをもとに配置される．リング上の各ノードは，ハッシュ値が自身のトークン値以下でかつ直前ノードのトークン値より大きい範囲を担当する．KVSの読み込みまたは書き込みをする際にはKeyをハッシュ関数にかけ，そのハッシュ値を担当するノードが読み込みや書き込みを実行するノードとなる．ただし，後述のレプリカが存在する場合，それを持つノードも実行するノードの対象となる．また，稼働中のシステムに新規ノードを追加する場合，新規ノードのトークン値から新規ノードの担当範囲が決まり，その範囲を担当している稼働ノードから担当範囲分のデータを取得する．稼働中のシステムからノードを動的に削除する場合，削除されるノードは担当している範囲のデータを新しく担当するノードに転送する．

Cassandra ではデータベースの複製(レプリカ)の数を指定することが可能である．レプリカ数は初期設定では1であるが，2以上の値にすることによって耐障害性を向上させることができる．レプリカは上記の担当ノードの後続ノードに配置される．

次に仮想ノード配置について述べる．仮想ノード機能はCassandra-1.2.0から実装されている．仮想ノード配置では，物理ノードが複数(初期設定では256個)の仮想ノードを持ち，各物理ノードは自身が持つ仮想ノードの担当範囲の合計を担当する．仮想ノードのトークン値はランダムに決められる．仮想ノードを用いることにより，物理ノードが数台しか存在しない環境であっても，ハッシュ空間上では仮想的に数百台のノード(仮想ノード)が存在していると認識させることが可能である．非仮想ノード配置では，ノード追加，削除時は(あるノードの担当範囲は連続している1つの領域であるため)データの追加や削除は特定のノードに対して集中的に行われることとなる．これに対して，仮想ノード配置では，ノード追加，削除時のデータの追加削除も，分散して行われ，均等に(あるいは偏りを持たせて)行うことができる．

3. ノード追加処理時間の測定

Cassandra は，システム稼働中にノードを追加することによって動的に性能を拡張することができる．本章では，Cassandra システムにおけるノード追加処理に要する時間(join 命令を発行した時刻から，システム状態が Joining から Normal に変わる時刻までの時間)の評価を行う．

評価環境は既存ノードの PC3 台，新規追加ノード用の PC1 台で構成される．すなわち，3 ノードで構成される KVS システムに 1 台のノードを追加し 4 ノードのシステムに変更する時間を測定する．仮想ノード配置を使用し，各ノードが持つ仮想ノード数は256である．測定はレコード数1600万件(約17[GB])のデータベース，レプリカ数3で行った．データベースの作成にはYCSB[3]を使用した．

A Study on Decreasing KVS Node Joining Time
Shohei Miyokawa†, Taiki Tokuda†, Saneyasu Yamaguchi†
†Electrical Engineering and Electronics, Kogakuin University Graduate School

3.1. KVS 実装とノード追加時間の関係

本節では Cassandra-2.0.0 と Cassandra-2.1.2 のノード追加時間の比較を行う．図 1 は Cassandra-2.0.0 と Cassandra-2.1.2 のノード追加処理にかかる時間を示している．図 1 から Cassandra-2.1.2 は Cassandra-2.0.0 よりノード追加時間が約 30 秒短いことがわかる．

図 2 は各ノードの Disk I/O 使用率を示している．図より，Cassandra-2.1.2 ではすべての既存ノードが Disk を使用しているが，Cassandra-2.0.0 では既存ノードのうち Node1 のみが Disk を使用していることがわかる．また新規ノード(Node4)の Disk I/O 使用率は Cassandra-2.1.2 が高く，Cassandra-2.0.0 の使用率の約 2.5 倍であることがわかる．

3.2. I/O スケジューラとノード追加時間

本節では I/O スケジューラとノード追加時間の関係について述べる．

図 3 はノード追加処理を I/O スケジューラの CFQ と Deadline で実行した際のノード追加時間である．Cassandra-2.0.0 は初期設定である CFQ から Deadline に変更することによるノード追加時間の短縮は見られなかった．Cassandra-2.1.2 は CFQ から Deadline に変更することによってノード追加時間が約 20 %短縮することがわかった．

図 4 は Cassandra-2.1.2 の新規ノードの Disk のシーク回数とシーク距離の関係をヒストグラムで示している．図 4 から 1[GB]以上のシーク距離のシーク回数は Deadline が CFQ よりも少ないことがわかり，これが Deadline の方がノード追加時間が短い理由と考えられる．

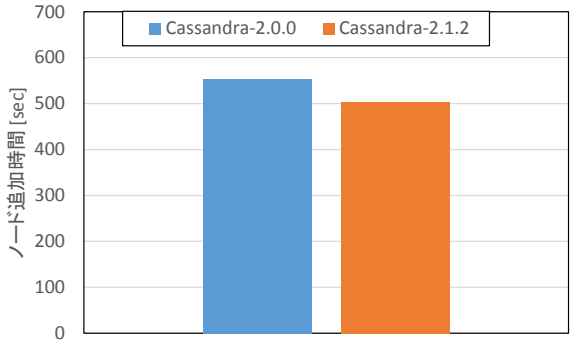

図 1. ノード追加時間(1)

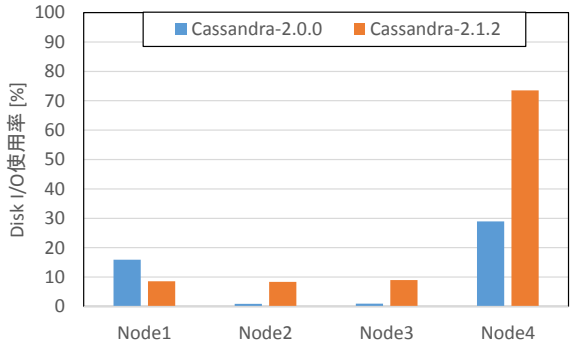

図 2. 各ノード Disk I/O 使用率

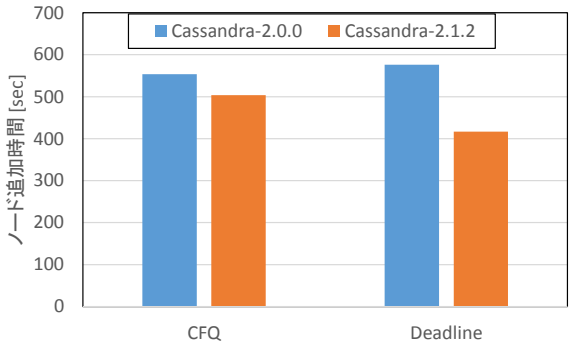

図 3. ノード追加時間(2)

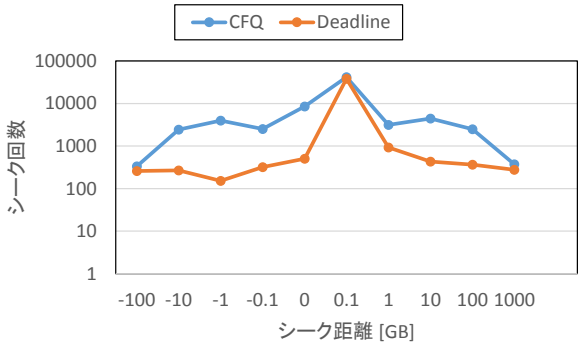

図 4. シーク回数とシーク距離の関係 (Cassandra-2.1.2)

4. おわりに

本研究では Cassandra のノード追加時間に着目し，Cassandra の実装や I/O スケジューラとノード追加時間の関係について考察した．今後は，既存ノード数によるノード追加性能への影響について考察する予定である．

謝辞

本研究は JSPS 科研費 24300034，25280022，26730040 の助成を受けたものである．

参考文献

[1] Facebook ページの投稿が一番多いのは、何曜日の何時？ ～2,000 社の Facebook ページを集計 http://comzenbunosez.com/?p=1219

[2]Avinash Lakshman and Prashant Malik, “Cassandra- A Decentralized Structured Storage System”, LADIS 09, (2009).

[3]Brian F. Cooper, Adam Silberstein, Erwin Tam, Raghu Ramakrishnan, Russell Sears, “Benchmarking Cloud Serving Systems with YCSB” ACM symposium on Cloud computing, (2010).

[4]堀内 浩基,山口 実靖, “KVS における動的性能伸張性の向上” 研究報告マルチメディア通信と分散処理（DPS）,Vol.2013-DPS-154(39),No.10 (2013)