1X-03

広域かつ大量に設置されたIoTデバイスが発するデータの管理を目的とした分散型データベースの提案

山内隆広† 宮崎敏明†

†会津大学コンピュータ理工学部

1. はじめに

近年、人々の健康管理や一人暮らしの見守りを目的に、センサなど様々なIoTデバイスを屋内に設置したスマートホームが注目されている。高齢化社会への対応として、上記スマートホームが全国規模で普及すれば、収集されるデータも膨大な量となるため、常に生成される大量のデータを効率よく保存・管理できるデータベースが必要となる。本稿では、IoTデバイスの一例として、安価なパッシブRadio Frequency Identification(RFID)が、日本全国規模で大量に設置されたと仮定し、分散データベースの一つであるApache Cassandra[1]を用いて地域ごとに分散配置した複数のデータベース管理システム(DBMS)を連携し、効率的に大量データを管理する分散型DBMSを提案する。

2. システム概要

本稿では、図1に示すように複数のRFIDタグが、壁や天井、家具類など家中に配備されたRFIDスマートホームが日本全国に配置されている状況を想定とする。各RFIDスマートホームでは、各RFIDタグから発せられた情報はRFIDリーダ（以下、単にリーダ）によって読み取られる。これをRFIDデータと呼ぶ。RFIDデータは、さらに各地域(ローカルエリア)に配置された近傍のデータベースノード（DBノード)に集約される。これらのDBノードは集合として全体のDBシステム、すなわち、DBクラスタを構成する。図2に、提案システムの全体構成を示す。複数のDBノードはネットワークで接続され、大きな分散DBシステムを構成する。

本システムでは、広域に分布するRFIDスマートホームから生成されるデータを管理するシステムであるため、突発的なDBノードの故障やネットワークの遮断が発生した際にもシステムが作動し続ける必要がある。また、ネットワーク遅延やDBノードの着脱性の観点から、RFIDタグから生成されたデータは最寄りのDBノードに保存されることが望ましい。RFIDスマートホームの規模が変動することでRFIDタグの数も増減する。ゆえに、必要な規模

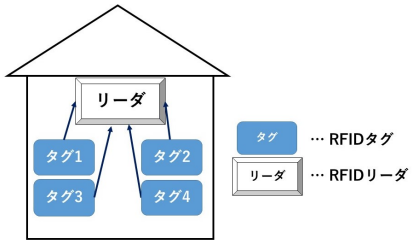

図1 RFIDスマートホーム

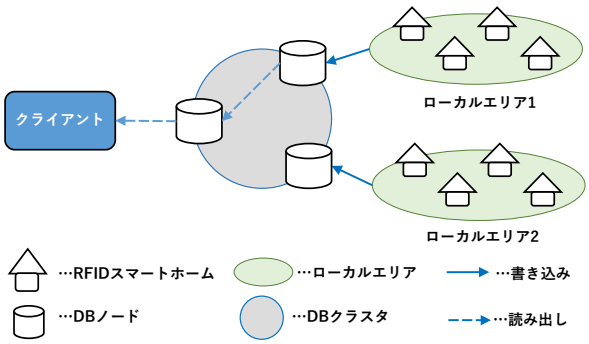

図2 提案システムの概要

に応じてシステムを停止することなくストレージ容量を拡大・縮小できる必要がある。

本稿では、10万軒のRFIDスマートホームの管理を当面の目標とし、1軒当たり平均30枚のRFIDタグが配備されていると仮定し、各RFIDタグから毎秒データが生成されるとする。すなわち、最大300万データ/秒の書き込み要求を満たすことを目標とする。

3. データ構造

図3に、本システムで用いたDBのテーブル構造をCQL(Cassandra Query Language)形式で示す。前述したようにデータの近傍ストアを実現するためには、キー値に地理情報を与えなければならない。ここでは、ローカルエリアの市町村コードを使用する

```
CREATE TABLE rfid.tag_to_data (
      code_id text,
      ip text,
      phase double,
      rssi double,
      doppler_freq double,
     time bigint,
     PRIMARY KEY (code_id, time)
) WITH CLUSTERING ORDER BY (time DESC)
```

図3 本システムで用いたデータベースのテーブル構造

Distributed database system to manage huge amount of data generated by widely deployed IoT devices
Takahiro Yamauchi†, Toshiaki Miyazaki†
†School of Computer Science and Engineering, The University of Aizu

ことで、それを実現する。tag_to_data 内のキーである code_id は6桁の市町村コードと8桁の RFID タグの固有 ID をつないだ文字列として保存される。また、Cassandra では、データ内のある要素でソートしてデータを保存することができる。これにより、時間による範囲検索が可能になる。

4. 評価および考察

Apache Cassandra は、キー・バリュー型の分散型データベースである。Cassandra を構成するノードは互いに P2P(Peer to Peer)通信している。Cassandra は書き込みデータが保存されるマシンの決定方法を複数から選択できる。本稿では、近傍 DB ノードにデータを格納する様にし、前述した近傍ストアを実現している。

ここでは、提案システムの基本性能の確認として、近傍ストアの設定をした Cassandra マシンを使用し、100 万データの書き込み、読み出し速度の計測を行った。計測には予め仮想マシン環境 XenServer[3]がインストールされている表 1 に示した仕様の物理マシン 2 台を用意し、それぞれに表 2 に示した仕様の仮想マシンを 1 台構築した。2 台の仮想マシンの内 1 台を Cassandra マシンとし、もう 1 台をクライアントとして評価環境を構築した。計測には Cassandra Stress[2]を使用した。Casandra は、複数のスレッドを用いて、複数のデータ書き込み・読み出し要求を同時に処理できる。

計測方法は下記のとおりである。まず、Cassandra マシン 1 台に対する書き込み件数の限界を調べるため、クライアントから Cassandra Stress を実行し、同時に実行できるスレッド数を 100 ずつ増加させながら、性能の変化が見られなくなるまで計測を続けた。それぞれのスレッドにつき、書き込み速度計測を 10 回行い、その平均を 1 秒当たりの処理件数とした。読み出し速度の測定では、予め 150 万データを書き込んでおき、そこから 100 万データを読み出すのに要した時間を、書き込み測定と同様に、スレッド数を変化させながら計測した。試行は、同一条件下で 10 回行い、その平均を当該条件下での読み出し速度とした。測定結果を図 4 に示す。図の縦軸の row とは、1 件の書き込み、または読み出し処理で取り扱われる、図 3 で定義した 1 連のデータ構造の単位である。1 台の Cassandra マシンでは、最大で毎秒 28137 件(1500 スレッド時)の書き込みができることが分かった。一方、読み出し速度は、最大で毎秒 13897 件(500 スレッド時)であった。これより、表 1 および表 2 に示したマシン環境では、目標である毎秒 3 百万データの書き込みを実現するためには 107 台の Cassandra マシンが必要であり、同量の読み出し件数を達成するためには 216 台の Cassandra マシンが必要となる。これは、実現可能なシステム規模である。

表 1 物理マシンのハードウェア環境

CPU	Intel® Xeon® CPU E3-1240 V2 @ 3.40GHz (4 コア 8 スレッド)
メモリ	8 GB
HDD	500 GB, 7200 rpm

表 2 仮想マシン環境

ハードウェア割り当て		ソフトウェア	
CPU	2	OS	CentOS 6.10
メモリ	4 GB	JDK	1.8.0
HDD	100 GB	Cassandra	3.0.9

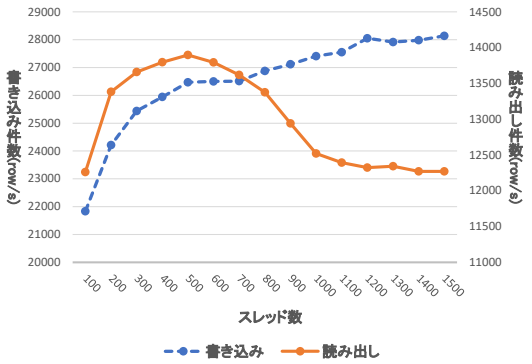

図 4 1 台の Cassandra マシンに対する平均書き込み・読み出し件数

また、図 4 から分かるように、スレッド数の増加に伴って、書き込み速度は一様に改善するが、読み出し速度に関しては、そうではない。これは、Casandra 内部には書き込みバッファが存在し、各スレッドによるデータ書き込みを同時に処理できる一方、読み出しに関しては、DB ノードのディスク性能が直接影響し、スレッド数を増やしても、1 つのディスクから同時に読み出せるデータに限りがあることからくる現象である。読み出し速度を改善するには、より高性能なディスクを用いるか、複数の DB ノードを用いて、1 つ当たりの DB ノードが受け持つ RFID スマートホームの数を減らす工夫が必要である。

5. おわりに

本稿では、日本全国規模に分散している RFID スマートホームを一括管理できるスケーラブルな DB システムを提案し、実験により RFID データ 300 万件の書き込み・読み出しを実現する分散型 DBMS の構築が可能である見通しを得た。

謝辞
本研究の一部は、総務省戦略的情報通信研究開発推進制度（SCOPE No.162302008）の支援を受けて実施したものである。

参考文献

[1] Apache Cassandra, http://cassandra.apache.org/
[2] Cassandra Stress, http://cassandra.apache.org/doc/4.0/tools/cassandra_stress.html
[3] Citrix XenServer, https://xenserver.org/