

推薦論文

IoT環境における処理削減による ストリーミング処理時間短縮手法

義久 智樹^{1,a)} 原 隆浩²

受付日 2016年6月8日, 採録日 2017年1月10日

概要: センサなどのモノがインターネットに接続される流行は, IoT (Internet of Things) と呼ばれ, 近年非常に注目されている. これらのモノから継続的に発生するデータを入力として, あらかじめ登録された問合せ (連続問合せ) を実行するストリーミング処理技術が研究されている. これらの研究では, データが入力されてから連続問合せの結果を得るまでのストリーミング処理にかかる時間を短縮している. 近年のIoT環境の普及により, 高頻度なデータの発生源が分散してインターネットに接続されており, さらなるストリーミング処理時間の短縮が求められている. そこで本研究では, IoT環境におけるストリーミング処理時間短縮手法を提案する. 提案手法では, センサが接続された処理サーバが分散していると想定したうえで, 各連続問合せを個別に処理できるいくつかの部分条件に分割してストリーミング処理を行う. 評価の結果, 提案手法は従来手法よりも最大通信ホップ数と平均通信量を削減でき, ストリーミング処理時間を短縮できることを確認した.

キーワード: センサ, 連続問合せ, インターネット, 分散処理

A Stream Processing Time Reduction Scheme by Process Reduction for IoT Environments

TOMOKI YOSHIHISA^{1,a)} TAKAHIRO HARA²

Received: June 8, 2016, Accepted: January 10, 2017

Abstract: Recently, IoT (Internet of Things) environments, in which various things including sensors connect to the Internet, have attracted great attention. To execute registered continuous queries for the data continuously generated from these things, various stream processing technologies have been studied. One of the main research issues for stream processing is how to reduce stream processing time, i.e., the elapsed time from getting data to getting query results. In IoT environments, many high frequency data sources distribute in Internet and further stream processing time reduction is required. Hence, we propose a stream processing time reduction scheme for IoT environments in this paper. Our proposed scheme focuses on distributed data sources with processing servers and extracts continuous queries to some conditional expressions. Processing servers can process each conditional expression individually. Our evaluation results show that our proposed scheme can reduce maximum communication hops and average communication traffic in IoT environments.

Keywords: sensor, continuous query, Internet, distributed processing

1. はじめに

センシング技術の発達とともに人感センサやカメラ

本論文の内容は2015年10月の第23回マルチメディア通信と分散処理ワークショップにて報告され, 同研究会主査により情報処理学会論文誌ジャーナルへの掲載が推薦された論文である.

¹ 大阪大学サイバーメディアセンター
Cybermedia Center, Osaka University, Ibaraki, Osaka 567-0047, Japan

² 大阪大学大学院情報科学研究科
Graduate School of Information Science and Technology,
Osaka University, Suita, Osaka 565-0871, Japan

^{a)} yoshihisa@cmc.osaka-u.ac.jp

などの様々なセンサがインターネットに多数接続されている。センサなどのモノがインターネットに接続される流行は、IoT (Internet of Things) と呼ばれ、近年非常に注目されている。これらのセンサには、設置されてから継続的にデータを計測しているセンサがあり、日々大量のデータがインターネット内で発生している。これらの継続的に発生するデータを入力として、あらかじめ登録された問合せ(連続問合せ)を実行するストリーミング処理技術が研究されている。たとえば、警備会社が契約先の1人暮らしの家の家電の利用状況をセンシングし、ふだんは利用するはずの複数の家電が利用されていなければ通知する連続問合せをストリーミング処理システムに登録する。家電が利用されていなければ、倒れていたり事件に巻き込まれて家に帰っていないといった可能性がある。また、人感センサでビルの人の出入りをセンシングしており、開錠時間外にセンサが反応した場合にその付近のカメラ映像の変化が激しければ人が侵入していると判断し、盗難の危険を発見することが考えられる。

データが入力されてから連続問合せの結果を得るまでのストリーミング処理にかかる時間を短縮することで、事件や火災の危険を即座に発見するなど応答性を高められる。また、ストリーミング処理時間を短縮することで、継続的に入力されるデータに対する単位時間あたりの最大の処理回数も向上できる。このため、ストリーミング処理時間を短縮する手法が提案されている(文献[1], [2], [3], [4], [5])。いくつかの手法では、データの発生源が複数の端末に分散している場合に、データのある端末に集約してから入力することで、個々の通信にともなうオーバーヘッドを削減している。オーバーヘッドの削減にともない、通信時間も削減できるため、ストリーミング処理時間を短縮できる。また、連続問合せを処理する処理サーバが複数ある場合に、処理負荷を分散させて連続問合せ処理を並列に行うことで処理にかかる時間を短縮してストリーミング処理時間を短縮する手法も提案されている。

一方で、近年のIoT環境の普及により、データの発生頻度が大きいデータ発生源が分散してインターネットに接続されており、さらなるストリーミング処理時間の短縮が求められている。上記したような警備会社の契約数は約90万件(文献[6])や約200万件(文献[7])ある。たとえば、100万件的契約先のビルから、1秒あたりにセンサデータを1個集めるとしても100万個のデータを処理することになる。人感センサ程度でデータサイズが比較的小さく、センサデータ1個あたり20バイト(IPパケットの最小データサイズ)の場合、160Mbpsのデータ発生頻度になる。映像データのようにデータサイズが比較的大きく、センサデータ1個あたり100Kバイト(MPEG形式の1フレーム相当)の場合では、800Gbpsのデータ発生頻度になる。ストリーミング処理時間が長くなると、遅延が重なってさら

に処理完了までの時間が長くなる。これまでの手法では、データを集約して通信時間を削減するか並列に処理して処理時間を短縮するアプローチを採用しており、IoT環境において上記の例のようにデータの発生頻度が大きい場合には、さらなるストリーミング処理時間の短縮が難しかった。センサが接続された処理サーバ(データの発生源と処理サーバ)が分散していると想定したうえで、冗長な処理を削減するアプローチにより、その処理にともなう通信時間も削減でき、ストリーミング処理時間をより短縮できる。

そこで本研究では、IoT環境において、センサが接続された処理サーバが分散してインターネットに接続されている点を考慮して、ストリーミング処理時間短縮手法を提案する。提案手法では、各連続問合せを個別に処理できるいくつかの部分条件に分割する。部分条件とは、温度データが異常に高い、カメラに炎が映っているといった連続問合せの各条件を指す。処理結果が満たされない確率の高い部分条件から順番に各処理サーバが処理し、部分条件が満たされない場合には残りの処理を省略する。2種類の提案手法において、部分条件の真偽判定を行うタイミングを決定して冗長な処理を削減できる点が新しく、ストリーミング処理時間の短縮に貢献できる。さらに、本研究では、提案手法の評価、議論を行う。

以下、2章で関連研究について説明し、3章で本研究で想定するIoT環境におけるストリーミング処理システムを説明する。4章でストリーミング処理時間短縮手法を提案し、5章で提案手法の評価、6章で考察を行う。最後に7章で本稿をまとめる。

2. 関連研究

文献[8]のように、発生したセンサデータを保存しておいて、後で連続問合せを行う場合、問合せ処理に必要なデータが保存されるまで時間がかかり、データの発生から処理結果を得るまでの時間が長くなる。連続的に発生するデータに対してストリーミング処理することで、素早く結果を得られるため、ストリーミング処理に関する研究が多数行われている。

文献[1]では、データの発生源とは異なる複数の処理サーバでストリーミング処理を行う場合に、処理時間を短縮できる処理割当て手法を提案している。センサデータのパラメタ空間を考慮して割り当てることで、パラメタが変化しても性能が劣化しにくい割当てを実現している。本研究において、後に述べる部分条件管理センサデータサーバの選定に本手法を利用できるが、我々の提案手法では、処理割当てではなく、冗長な処理を削減することでストリーミング処理時間を短縮している点が異なる。

Lohrmannらは文献[2], [3]において、ストリーミング処理時間を制限以内に収める手法を提案している。処理時間を予測し、可能な限り制限時間内に完了できるように処

理を並列化している．後に述べるトリガグラフの作成に Lohrmann らの手法を利用できるが，本研究の提案手法では，連続問合せをいくつかの処理に分割して冗長な処理を削減することでストリーミング処理時間を短縮している点が異なる．

Zhang らは文献 [9] において，複数のデータストリームを集約しながら 1 本のデータストリームにまとめることで，通信回数を削減する手法を提案している．Zhang らの手法もトリガグラフの作成に応用できるが，本研究では，連続問合せ処理においてストリーミング処理時間の短縮を目的としている点が異なる．

文献 [4] では，ストリーミング処理において，Streaming Quotient Filter (SQF) と呼ぶデータ構造を用いて重複する処理結果を 1 個にまとめて通信量を削減する手法を提案している．また，文献 [5] では，複数のデータストリームの結合処理において，過去の複数のデータをスケッチとして高速な記憶領域に貯えておくことで，結合処理を高速に行える手法を提案している．処理するデータに対してこれらの手法を用いることで，ストリーミング処理時間を短縮できる．

3. 想定環境

本章では，本研究で想定している IoT 環境におけるストリーミング処理システムを明確にし，システム構成などについて説明する．

3.1 部分条件

本節では，連続問合せに含まれる条件の部分条件への分割方法を説明する．

N 個の連続問合せを Q_n ($n = 1, \dots, N$) で示す．一般に，連続問合せは記述された複数の条件 (式だけでなく射影や選択も含まれる) が満たされた場合に実行結果をユーザに提示する．条件が満たされるかどうかの真偽は論理関数で求められるため， Q_n が和積標準形の論理関数で表現できると考えても一般性を損なわない．射影や選択演算であっても，ある和項に含めることで和積標準形で表現できる．具体的な例は次節で説明している．この和積標準形で記述された Q_n の条件の I_n 個の和項を $C(n, i)$ ($i = 1, \dots, I_n$) で示し部分条件と呼ぶ．部分条件が満たされて真偽判定結果が真の場合 1，偽の場合 0 とする．下記が満たされると Q_n の実行結果がユーザに提示されることになる．

$$\prod_{i=1}^{I_n} C(n, i) = 1 \quad (1)$$

3.2 システム構成

クライアントは，ユーザからの連続問合せを受け，その実行結果をユーザに提示する．センサデータサーバは複数あり，各センサデータサーバにはセンサが接続されてい

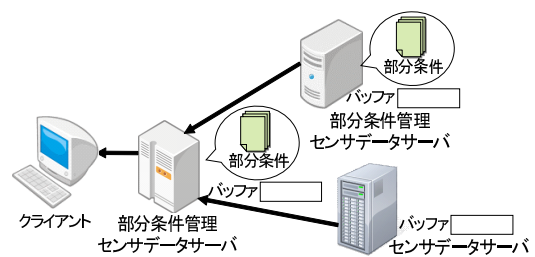


図 1 システム構成の一例

Fig. 1 An example of the system image.


る．各センサデータサーバは，センサからセンサデータを取得し，センサデータや部分条件の真偽判定結果を送信するためのバッファを備えている．本論文では，部分条件の真偽判定を行うセンサデータサーバを部分条件管理センサデータサーバと呼ぶ．各部分条件管理センサデータサーバは，部分条件の真偽判定に必要なセンサデータを，それらを持つセンサデータサーバから取得して部分条件の真偽判定を行う．クライアントおよび各センサデータサーバはインターネットなどの情報ネットワークで接続されており，互いに通信可能とする．IoT 環境におけるシステム構成の一例を図 1 に示す．以上のシステム構成において，下記の環境を想定する．

- センサデータサーバは，センサから継続的にデータを取得する．
- クライアントは IP アドレスなどのセンサデータサーバの識別子を取得している．
- クライアントは，登録された連続問合せを部分条件に分割できる．
- 部分条件の実行に必要なセンサデータはいずれかのセンサデータサーバが持っている．

具体例として，1 章に記述したとおり，警備会社が多数の契約先のビルに設置したセンサからセンサデータを取得し，複数のセンサデータサーバを用いてストリーミング処理を行う場合が考えられる．後の評価では，この例を参考にして評価環境を設定している．

$C(n, i)$ の真偽判定に必要なセンサデータを持つセンサデータサーバの識別子の集合を $Servs(n, i)$ で示し， $Servs(n, i)$ に含まれる， $C(n, i)$ の部分条件管理センサデータサーバを $MServ(n, i)$ で示す． $MServ(n, i)$ は， $C(n, i)$ の真偽を判定するために， $Servs(n, i)$ に含まれる複数のセンサデータサーバからセンサデータを取得する． $p, q \in Servs(n, i)$ に関して， $C(n, i)$ の真偽判定のために必要な p から q への通信の頻度を $ComF_{p \rightarrow q}(n, i)$ で示す．センサデータが連続的に発生するため，本研究では通信の頻度を用いる．また， $C(n, i) = 1$ となる確率を $TProb(n, i)$ で示す． $TProb(n, i)$ の予測値は，システムメンテナンス時や定期的に，クライアントが $C(n, i)$ の真偽を判定したセンサデータサーバから， $C(n, i) = 1$ となった割合を収集

センサデータサーバ



センサデータ	時刻	部分条件	データ
Aの温度データ	12:02	1, 2, 5	20
Aの映像のデータ	12:02	1, 2, 5	a.jpg
Bの温度データ	12:00	3, 4, 5	80
Bの映像のデータ	12:00	3, 4, 5	b.jpg

図 2 センサデータサーバ A が保持するセンサデータ
 Fig. 2 Sensor data server A's sensor data.

することで算出できる。

連続問合せの条件を和積標準形で表現したとき、あるセンサデータサーバが取得しているセンサデータが複数の和項（部分条件）に含まれている場合、当該センサデータサーバは複数の *Servs* に含まれる。各センサデータサーバはセンサデータを連続的に取得しており、各部分条件の真偽判定には保持する最新のセンサデータを用いる。

3.3 連続問合せの例

警備会社が火事を素早く発見するために、温度センサとカメラを接続したセンサデータサーバを用意する。温度が高くカメラに炎が映っていれば火事と判断し、そのときの温度と画像をユーザに提示して注意喚起する。誤認識を防ぐために、センサデータサーバを2台 (AとB) 用意し、これらを近い場所に設置してセンシングする。上記条件に加えてセンサデータサーバ A, B の温度データの差が小さい場合に火事と判断する。温度センサの計測頻度は 1Hz, カメラの撮影頻度は 30 Hz とする。一例として、上記のセンサデータサーバ A が図 2 に示すような形式でセンサデータを保持することが考えられる。

この場合、連続問合せは $N = 1$ 個で、 Q_1 の部分条件は $I = 5$ 個になってそれぞれ、 $C(1,1)$ 「センサデータサーバ A の温度データが高い」、 $C(1,2)$ 「センサデータサーバ A のカメラに炎が映っている」、 $C(1,3)$ 「センサデータサーバ B の温度データが高い」、 $C(1,4)$ 「センサデータサーバ B のカメラに炎が映っている」、 $C(1,5)$ 「センサデータサーバ A と B の温度データの差が小さい」となる。 $C(1,1), C(1,2)$ の真偽の判定に必要なセンサデータを持つセンサデータサーバは A のみであるため $Servs(1,1)$ および $Servs(1,2)$ は $\{A\}$ 、また、 $Servs(1,3)$ および $Servs(1,4)$ は $\{B\}$ 、 $Servs(1,5) = \{A, B\}$ となる。 $C(1,5)$ の判定を A で行う場合 ($MServ(1,5) = A$)、B の温度データを A に送信する。センサデータサーバ A は、 $C(1,5)$ の部分条件管理センサデータサーバになる。単純につねに部分条件を判定し、B が温度データを計測するごとにセンサデータを A に送信する場合、 $ComF_{A \rightarrow B}(1,5) = 0$ 、 $ComF_{B \rightarrow A}(1,5) = 1$ となる。

4. 提案手法

式 (1) より、いずれかの部分条件が満たされずに真偽判

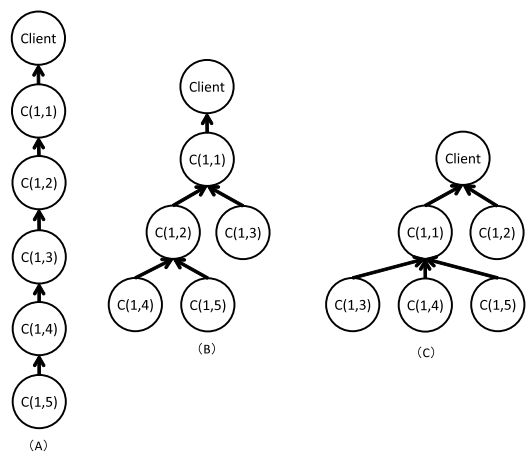


図 3 単純なトリガグラフの例
 Fig. 3 Examples for simple trigger graph.

定結果が偽であれば連続問合せの条件は満たされない。満たされない場合には、他の部分条件の真偽判定処理を削減でき、処理時間とその処理にともなう通信時間を削減できる。提案手法では、連続問合せ Q_n ($n = 1, \dots, N$) に関して、 $TProb(n, i)$ ($i = 1, \dots, I_n$) が小さく真になる確率が小さい部分条件から順番に真偽の通知を行う。すなわち、 $TProb(n, i_1) < \dots < TProb(n, i_I)$ とすると、 $C(n, i_1), \dots, C(n, i_I)$ の順番に通知する。 $MServ(n, i_j)$ ($j = 1, \dots, I-1$) は $C(n, i_j) = 1$ であれば、 $MServ(n, i_{j+1})$ に通知する。そこで、提案手法では、トリガグラフを用いて部分条件の真偽判定結果を通知する順序を決定する。さらに、トリガグラフにおける真偽判定のタイミングを2種類提案する。

4.1 部分条件のトリガグラフ

トリガグラフとは、ある部分条件が満たされた場合に次に真偽判定結果を通知する（トリガをかける）部分条件を示すグラフである。連続問合せごとに作成され、作成には文献 [2], [3], [9] の手法を利用できる。連続問合せの実行結果を通知するクライアントが根となり、部分条件がノードの木構造のトリガグラフとなる。一般的なトリガグラフにおいて、繰返して同じ処理を行う場合などにループを発生する構造になるが、ループ部分と同じノードをトリガグラフに追加することで、木構造のトリガグラフになる。本研究で用いる、積和標準形の和項をノードとしたトリガグラフも木構造になる。最大通信ホップ数はトリガグラフの形状から算出できる。平均通信量はトリガグラフと $ComF$ 、 $TProb$ から算出できる。真偽判定の順番に制約がなければ、ストリーミング処理を行う様々な形状のトリガグラフが考えられる。

たとえば、3.3 節の例の場合、クライアントおよび $C(1, i)$ ($i = 1, \dots, 5$) がノードとなり、単純なトリガグラフとして図 3 (A) が考えられる。図 3 (A) では、 $C(1,5)$ が満た

されていれば $C(1,4)$ にトリガをかけ、 $C(1,4)$ の部分条件管理センサデータサーバは部分条件の真偽判定を行い、 $C(1,4)$ が満たされていればさらに $C(1,3)$ にトリガをかけるといったように、順番にトリガをかける。グラフの深さは5である。また、図3(B)のようなトリガグラフも考えられる。図3(B)では、根以外で二分木を構成している。この場合のグラフの深さは3になる。さらに、図3(C)のようなトリガグラフも考えられる。後の評価で示すが、深さが大きいほど、削減できる真偽判定処理が多くなって通信量を削減できる傾向にあるが、最大通信ホップ数で示されるストリーミング処理時間は部分条件の真偽判定処理のタイミングによって長短が異なる。

部分条件が満たされた場合、深さが大きく葉に近い部分条件から根の方に向かってトリガがかけられるため、真偽判定結果が真になる確率の小さい部分条件を葉の方に配置することで通信量を削減できる。そこで、提案手法では、真になる確率の小さい部分条件をトリガグラフの葉の方へ配置したトリガグラフを作成している。

4.2 真偽判定のタイミング

部分条件の真偽を判定するタイミングによってストリーミング処理時間が異なる。本研究では、真偽判定するタイミングとして、以下の2種類を考える。

4.2.1 独立判定法

独立判定法では、各部分条件管理センサデータサーバは他の部分条件の真偽判定結果にかかわらず、部分条件の真偽を判定する。すなわち、 $MServ(n, i_j)$ は、自身のセンサデータが発生するたびに、受信している最新の $Servs(n, i_j)$ のセンサデータを用いて $C(n, i_j)$ の真偽を判定する。各センサデータサーバは、部分条件管理センサデータサーバに、センサデータが発生する度にセンサデータを送信する。センサデータが発生する度に通信が発生するが、あらかじめ真偽を判定しているため、次の逐次判定法と比べてトリガグラフの親ノードに真偽判定結果を通知するまでの時間を短くできる。

4.2.2 逐次判定法

逐次判定法では、各部分条件管理センサデータサーバはトリガをかけられてから部分条件の真偽を判定する。すなわち、 $MServ(n, i_j)$ は、トリガをかけられると $Servs(n, i_j)$ からセンサデータを取得して $C(n, i_j)$ の判定を行う。各センサデータサーバは、部分条件管理センサデータサーバから取得のメッセージが送信された場合にセンサデータを送信する。独立判定法と比べて通知するまでの時間が長くなるが、 $MServ(n, i_j)$ はトリガをかけられてから部分条件の真偽を判定するため、通信量を削減できる。

各部分条件管理センサデータサーバは、自身の真偽判定が偽であれば他の部分条件の真偽判定結果を確認する必要がないため、トリガをかけられると、自身の真偽判

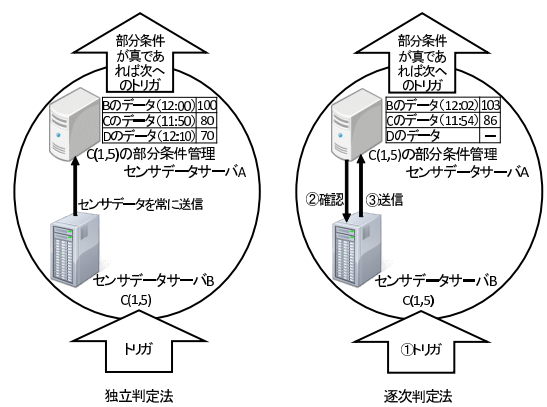


図4 処理開始のタイミング

Fig. 4 Timing for starting sensor data processing.

定を行い、真であれば、トリガグラフ上で、トリガをかけられた子ノード以外の子ノードの部分条件の真偽判定結果を確認する。確認の回数を確率的に減らすため、子が複数ある場合、真になる確率が低い部分条件の部分条件管理センサデータサーバから順番に確認を行う。すなわち、 $C(n, i_j)$ の K 個の子ノードが $C(n, k_1), \dots, C(n, k_K)$ で $TProb(n, k_1) < \dots < TProb(n, k_K)$. $C(n, k_1)$ が $C(n, i_j)$ にトリガをかける子ノードとすると、 $C(n, k_2), \dots, C(n, k_K)$ の順に確認を行う。提案手法では、トリガグラフの作成時に親ノードにトリガをかける子ノードを決定し、他の子ノードは確認されたときのみ真偽を親ノードに通知することで、親ノードが異なる複数の子ノードからトリガをかけられることを防ぐ。冗長な通信を削減するため、真偽判定結果が偽であれば、他の子ノードへの確認を中止する。

4.3 具体例

各方法の処理開始のタイミングを図4を用いて説明する。3.3節の例と同様に部分条件 $C(1,5)$ の部分条件管理センサデータサーバ $MServ(1,5)$ がAで、 $Servs(1,5) = \{A, B\}$ の場合を考える。図4では、センサデータサーバAが部分条件管理センサデータサーバであることを明確にするために部分条件管理センサデータサーバAと記載している。

独立判定法では、センサデータサーバBがセンサデータが発生する度に部分条件管理センサデータサーバAにセンサデータを送信する。部分条件管理センサデータサーバAが他の部分条件の判定のためにセンサデータサーバCやDからセンサデータを受信している場合には、それらの最新のセンサデータを保持している。部分条件管理センサデータサーバであるAがトリガグラフに沿ってトリガをかけられると、保持しているセンサデータを用いて部分条件を判定し、結果が真であれば次の部分条件へのトリガをかける。

逐次判定法では、部分条件管理センサデータサーバAがトリガをかけられると、 $Servs(1,5)$ に含まれる自身以外のセンサデータサーバに確認を行い、最新のセンサデータを受信する。部分条件管理センサデータサーバAが保持する

センサデータは、トリガをかけられた時点で各センサデータサーバが保持する最新のセンサデータとなる。部分条件管理センサデータサーバである A が部分条件の判定に必要なすべてのセンサデータを受信すると、部分条件を判定し、結果が真であれば次の部分条件へのトリガをかける。

5. 評価

本章では、提案手法の評価を行う。文献 [10], [11], [12] といったセンサデータが公開されているが、ストリーミング処理の連続問合せ対象となるセンサデータは様々であるため、ここでは評価用に人工的に発生させたストリーミングデータを用いる。提案手法はストリーミング処理時間の短縮を目的としているが、ストリーミング処理時間はセンサデータサーバの処理能力や通信速度に依存する。具体的な処理速度と通信遅延およびいくつかの仮定をおくことで、処理時間を算出できるが、これらの値や仮定は処理基盤に依存し、実環境では変化することも考えられる。本評価では、処理能力や通信速度といった処理基盤の影響を除いて評価を行うため、最大通信ホップ数および平均通信量で提案手法を評価する。後に 6.1 節で詳述するが、これらを抑えることでストリーミング処理時間を短縮できる。

最大通信ホップ数には、センサデータサーバ間の通知にかかるホップ数も含まれており、たとえばトリガグラフが図 3(B) (評価で用いる列トポロジで深さ 3, 枝の数 2) で各部分条件のセンサデータサーバが 1 個の場合、独立判定法では、葉の部分条件管理センサデータサーバがセンサデータサーバからセンサデータを受信するのに 1 ホップ、根の部分条件管理センサデータサーバまで通知するのに 2 ホップ、クライアントに通知するのに 1 ホップで最大通信ホップ数は 4 ホップになる。逐次判定法では、各部分条件管理センサデータサーバがセンサデータサーバにセンサデータを要求して取得するのに 2 ホップずつかかり、隣接する各部分条件管理センサデータ 2 点間の通知の合計で 6 ホップ ($C(1,4)$ から $C(1,2)$ への通知に 1 ホップ、 $C(1,2)$ の $C(1,5)$ への真偽判定結果の要求と取得で 2 ホップかかり、さらに $C(1,2)$ から $C(1,4)$ への通知に 1 ホップ、 $C(1,4)$ の $C(1,3)$ への真偽判定結果の要求と取得で 2 ホップ)、クライアントへの通知に 1 ホップかかるため、最大通信ホップ数は 17 ホップになる。

本評価では、提案手法をトリガグラフを用いない全集約法および順次集約法と比較する。全集約法では、ある部分条件管理センサデータサーバにすべてのセンサデータを集約してから連続問合せを実行する最も単純な手法である。全集約法では、ブロードキャストアドレスなどを用いて並行してセンサデータを要求できない場合にも適用できる汎用的な手法とし、最も単純な手法の代表とするためにも、各センサデータサーバに逐次的にセンサデータを要求している。順次集約法では、全集約法と同様にある部分条件管

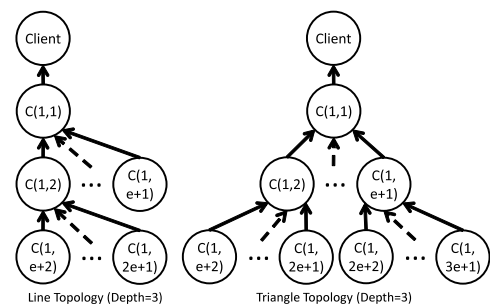


図 5 評価に用いたトリガグラフのトポロジ

Fig. 5 Topologies of trigger graphs for evaluation.

理センサデータサーバが中心となって順次データを集約していくが、真偽判定結果が偽になれば集約を中止する。すなわち、順次集約法では、単純に真になる確率の低い部分条件から順番に真偽判定を行い、真偽判定結果が偽であれば、残りの部分条件の真偽判定を省略する。

5.1 評価環境

文献 [2], [3], [9] の手法を用いてトリガグラフを作成できるが、トリガグラフの作成手法ではなく、提案手法のアプローチを評価するため、本評価では図 5 に示す 2 種類の単純なトリガグラフを用いる。列トポロジ (Line Topology) では、根以外で各深さの部分条件の数が等しい。各評価で枝の数 e を変化させて評価を行う。三角トポロジ (Triangle Topology) では、葉以外で各部分条件に同じ数の子の部分条件がある。列トポロジと同様に枝の数 e を変化させて評価を行う。各部分条件のセンサデータサーバは、部分条件管理センサデータサーバの他に 1 台あるとした。IoT 環境の通信量の一例として、1 章で述べた警備会社の契約数を参考にし、独立判定法では、各センサデータサーバは 1 秒で 1M メッセージを部分条件管理センサデータサーバに送信する。各部分条件管理センサデータサーバも、1 秒で 100 万 (1M) メッセージをトリガグラフの親ノードに送信することになる。逐次判定法では、各センサデータサーバは部分条件管理センサデータサーバが確認した時のみセンサデータを部分条件管理センサデータサーバに送信する。トリガやセンサデータ、真偽判定の結果は 1 メッセージで送受信できるとし、真偽判定を開始する葉ノードも 1 秒で 1M メッセージを送信するものとした。 $TProb(n, i)$ ($n = 1, \dots, N, i = 1, \dots, I_n$) は、評価で変化させる場合を除き、0.7 とした。

5.2 列トポロジ

枝の数と深さを変化させ、列トポロジの最大通信ホップ数をコンピュータシミュレーションにより計測した。結果を図 6 に示す。横軸は部分条件の数であり、たとえば枝の数が 2, 深さが 3 のとき 5 となる。縦軸は最大通信ホップ数であり、逐次判定法では、真偽判定を開始する

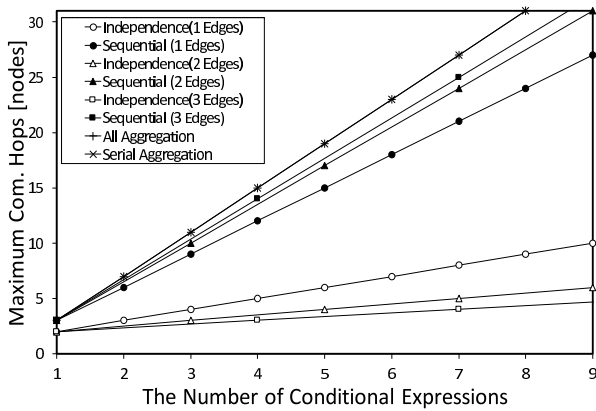


図 6 列トポロジの最大通信ホップ数

Fig. 6 Maximum communication hop for line topology.

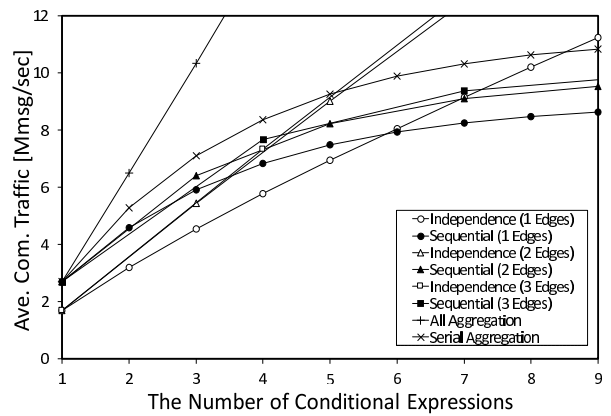


図 7 列トポロジの平均通信量

Fig. 7 Average communication traffic for line topology.

ノードからクライアントに結果が通知されるまでの最大の通信ホップ数となる。凡例中、‘Independence’ および ‘Sequential’ が提案する独立判定法および逐次判定法を示す。‘All Aggregation’ および ‘Serial Aggregation’ は比較手法の全集約法および順次集約法である。比較手法はトリガグラフを用いないため、枝の数を定義できない。

この結果より、独立判定法が、逐次判定法よりも最大通信ホップ数を小さくできることが分かる。これは、独立判定法では、各部分条件管理センサデータサーバが、センサデータが発生する度に真偽判定しており、他の部分条件管理センサデータサーバからトリガをかけられるとすぐに上位ノードに通知を送信できるためである。一方、逐次判定法では、トリガをかけられてから真偽判定するため、真偽判定に必要なセンサデータサーバとの通信が発生し、最大通信ホップ数が大きくなる。また、部分条件の数に比例して真偽判定のための通信が発生して最大通信ホップ数が大きくなっている。比較手法である全集約法および順次集約法では、ある部分条件管理センサデータサーバが、各部分条件管理センサデータサーバに部分条件の真偽判定結果を確認する。提案手法のようにトリガグラフを用いておらず、最大ですべての部分条件管理センサデータサーバと通信する（各2ホップ）ため、最大通信ホップ数は最も大きくなっている。

独立判定法では、部分条件の数と同じであれば枝の数が増えるほど深さが浅くなり、最大通信ホップ数が減少する。逐次判定法では、枝の数が増えるほど木構造の兄弟ノードとなる部分条件が多くなり、センサデータ取得のための通信が多くなって最大通信ホップ数も大きくなる。

次に、通信量を図7に示す。横軸は部分条件の数、縦軸は平均通信量であり、逐次判定法では、 $TProb$ を考慮したうえで平均的な通信量を示している。

この結果より、部分条件の数が少ない場合には独立判定法の平均通信量が逐次判定法の平均通信量よりも少ないことが分かる。これは、逐次判定法では、真偽判定に必要な

センサデータをセンサデータサーバに要求する際、1メッセージの通信が発生するためである。独立判定法では、継続的に部分条件管理センサデータサーバにセンサデータが送信されているため、この逐次判定法のような通信が発生しない。しかし、部分条件の数が多くなると、逐次判定法の平均通信量が独立判定法の平均通信量よりも少なくなっている。これは、各部分条件管理センサデータサーバで部分条件が満たされない場合に、トリガグラフで上位のノードにおいて部分条件を判定する必要がなく、これにともなう通信が発生しないためである。枝の数が多いほど発生する通信も増加するため、通信量が多くなる。

全集約法では、提案手法のように部分条件の真偽判定結果が偽であれば通信を行わないといった工夫をしていないため、平均通信量が最も多くなっている。順次集約法では、真偽判定結果が偽であれば残りの部分条件の真偽判定結果の確認を省略しているため、逐次判定法と同様の変化を示している。しかし、ある部分条件管理センサデータサーバに集約する分、逐次判定法より平均通信量が多くなっている。順次集約法では、独立判定法より平均通信量が少なくなる場合があるが、最大通信ホップ数が大きい。

たとえば、枝の数が2、深さが2で部分条件が3個の場合、独立判定法の最大通信ホップ数は3、平均通信量は5.4Mメッセージ/秒、逐次判定法の最大通信ホップ数は10、平均通信量は6.4Mメッセージ/秒となり、独立判定法の最大通信ホップ数、平均通信量ともに逐次判定法よりも小さい値を与えている。部分条件の数が増えて5（深さ3）の場合、独立判定法の最大通信ホップ数は4、平均通信量は9.0Mメッセージ/秒、逐次判定法の最大通信ホップ数は17、平均通信量は8.2Mメッセージ/秒となり、独立判定法の最大通信ホップ数が逐次判定法よりも小さくなるが、平均通信量は逐次判定法の方が少なくなる。逐次判定法の方が平均通信量が少なくなる部分条件の数は、トリガグラフの形状に依存する。

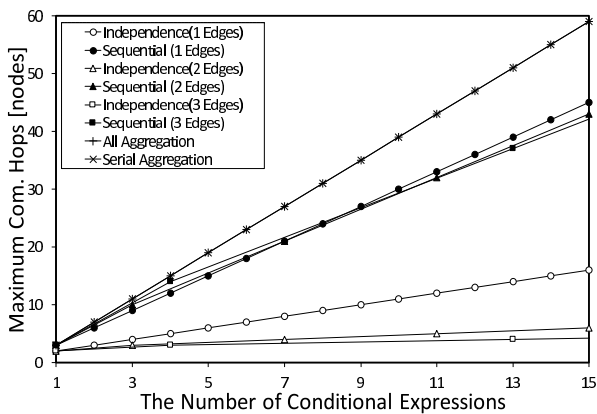


図 8 三角トポロジの最大通信ホップ数

Fig. 8 Maximum communication hop for triangle topology.

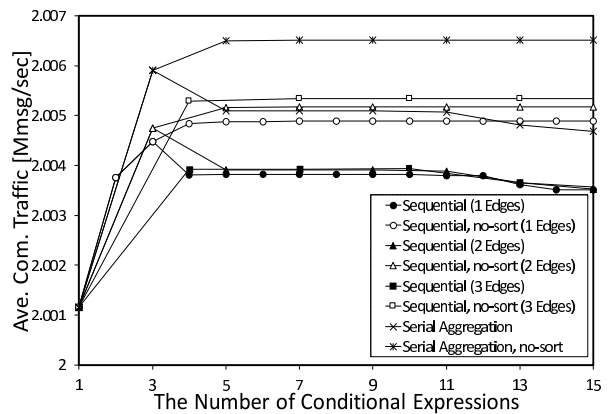


図 10 真になる確率の考慮の有無による差異

Fig. 10 Difference caused by true probability consideration.

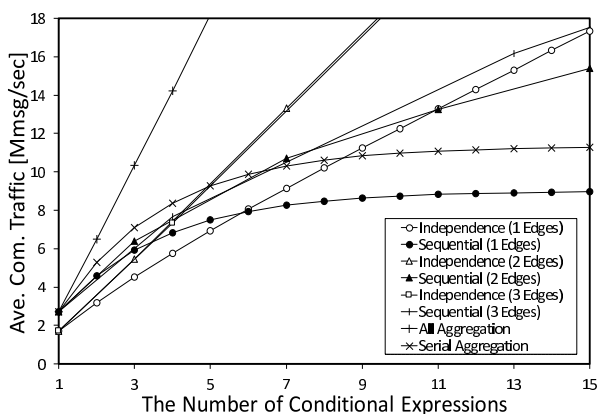


図 9 三角トポロジの平均通信量

Fig. 9 Average communication traffic for triangle topology.

5.3 三角トポロジ

各深さの枝の数と深さを変化させ、三角トポロジの最大通信ホップ数をコンピュータシミュレーションにより計測した。結果を図 8 に示す。横軸は部分条件の数であり、たとえば枝の数が 2、深さが 3 のとき 7 となる。縦軸は最大通信ホップ数である。

この結果より、独立判定法が逐次判定法よりも最大通信ホップ数を小さくできることが分かる。これは、独立判定法では、部分条件管理センサデータサーバからトリガをかけられるとすぐに上位ノードに通知を送信できるが、逐次判定法では、トリガをかけられてから部分条件を判定しているためである。列トポロジと同じく、独立判定法では枝の数が増えるほど最大通信ホップ数が小さくなり、逐次判定法では増加している。提案手法と比較手法との議論は、列トポロジと同様であるため省略する。

次に、通信量を図 9 に示す。横軸は部分条件の数、縦軸は平均通信量であり、列トポロジの場合と同じく、逐次判定法では、 $TProb$ を考慮したうえで平均的な通信量を示している。

この結果より、部分条件の数が少ない場合には独立判定法の平均通信量が逐次判定法の平均通信量よりも少ないこ

とが分かる。これは、逐次判定法では、判定に必要なデータをセンサデータサーバに要求する際、1 メッセージの通信が発生しているためである。列トポロジと同じく、枝の数が多いほど発生する通信も増加するため、平均通信量が多くなっている。

5.4 真になる確率を考慮する効果

提案手法では、 $TProb$ が小さい部分条件をトリグラフの葉の方に配置させることで、通信量を削減させている。この効果を調べるために、提案手法とランダムに配置する手法を比較した。独立判定法については、真偽判定の通知を定期的に行っていて $TProb$ の影響がないため、逐次判定法で調べた。真になる確率は最大通信ホップ数に影響を及ぼさないため、トリグラフの形状が同じであれば、配置が変わっても変化はない。

通信量をシミュレーションで求めた結果を図 10 に示す。横軸は部分条件の数、縦軸は平均通信量であり、'no-sort' が $TProb$ の値を考慮しない場合である。比較手法の 1 つである全集約法では、すべての部分条件の真偽判定結果を確認しており、真になる確率を考慮しないため結果に示していない。結果を見やすくするため、縦軸の最小値を 2、最大値を 2.007 としている。トリグラフは列トポロジを用いた。

この結果より、真になる確率を考慮することで、平均通信量を削減できることが分かる。これは、連続問合せの条件が満たされない場合には、センサデータを送信しないことで、通信量を削減できるためである。部分条件の数が 12 より多くなると提案手法で平均通信量が少なくなっているのは、 $TProb$ をランダムに与えており、 $TProb$ の最小値が小さくなったためである。

5.5 真の確率の影響

部分条件が真になる確率が、通信量に影響を与えるため、 $TProb$ を変化させて通信量をコンピュータシミュレーション

表 1 図 11 の最大通信ホップ数

Table 1 Maximum communication hop for Fig. 11.

手法	独立判定	逐次判定	全集約	順次集約
最大通信ホップ数	4	17	19	19

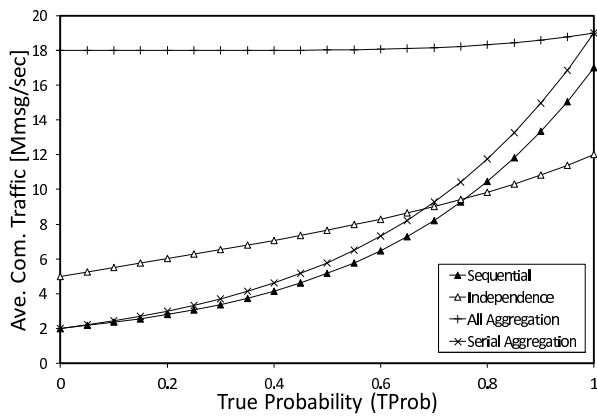


図 11 真になる確率と通信量

Fig. 11 Average communication traffic and $TProb$.

により求めた。トリガグラフは列トポロジを用い、枝の数は 2、深さは 3 とした。各手法の最大通信ホップ数を表 1 に示す。最大通信ホップ数は、 $TProb$ の影響はなく、トリガグラフの形状にのみ依存する。このため、通信量の結果を図 11 に示す。横軸は $TProb$ の値、縦軸は平均通信量である。

この結果より、真になる確率が小さい場合には逐次判定法の平均通信量が独立判定法より少なくなっていることが分かる。これは、前節で説明したとおり、逐次判定法では、トリガグラフの上位ノードにおける通信を削減できるためである。真になる確率が大きくなると、逐次判定法では、センサデータ取得のためのメッセージが必要なため、独立判定法に比べて平均通信量が多くなっている。

全集約法では、すべての部分条件の真偽判定結果を確認しており、真になる確率の影響を大きく受けないが、すべての部分条件が真の場合にクライアントへの通知にともなう通信量が発生するため、真になる確率が大きいほど平均通信量は他の手法と比べてわずかだが増加している。順次集約法では、逐次判定法と同様の変化を示しているが、トリガグラフを用いていないため、表 1 に示すとおり最大通信ホップ数が逐次判定法と比べて大きくなっている。

6. 考察

6.1 ストリーミング処理時間

本研究では、最大通信ホップ数および平均通信量で提案手法を評価した。最大通信ホップ数が小さくなるほど、通信遅延が短くなってセンサデータが発生してからストリーミング処理を完了するまでの時間が短くなり、ストリーミング処理時間を短縮できる。また、平均通信量が少なくなるほど通信にかかる時間が短くなってストリーミング処理

時間を短縮できる。通信帯域が小さかったり通信環境が不安定で遅延や再送が発生する場合には、平均通信量を少なくすることで、これらが発生する頻度が小さくなってストリーミング処理に時間がかかるのを抑えることができる。

具体的に求められるストリーミング処理時間は応用などによって異なるが、1章で述べた事件発見などの例では、短くすることで即座に発見できる。提案手法を用いることでストリーミング処理時間を短くでき、これらの応用に有効である。

6.2 列トポロジと三角トポロジ

いくつかのトリガグラフの形状で評価を行うため、本研究では列トポロジと三角トポロジを用いた。本節では、これらのトポロジの比較議論を行う。図 6～図 9 に示すとおり、部分条件の数が同じ場合、列トポロジより三角トポロジの方がトリガグラフの深さを抑えられるため、独立判定法において最大通信ホップ数を小さくできる。逐次判定法では各部分条件管理センサデータサーバが逐次的に真偽判定を行うため、大きくなる。平均通信量については、三角トポロジでは、子の部分条件管理センサデータサーバへの真偽判定の確認回数が多くなるため、列トポロジよりも多くなる。

7. まとめ

本研究では、IoT 環境において、センサが接続された処理サーバが分散してインターネットに接続されている点を考慮し、ストリーミング処理時間短縮手法を 2 種類提案した。提案手法では、各連続問合せを個別に処理できるいくつかの部分条件に分割し、処理結果が真になる確率が小さい部分条件から順番に処理する。独立判定法では、各部分条件管理センサデータサーバは他の部分条件の真偽判定結果にかかわらず、部分条件の真偽を判定する。逐次判定法では、下位のノードが真になる場合のみ部分条件の真偽を判定する。評価の結果、提案手法を用いることで最大通信ホップ数と平均通信量を削減でき、ストリーミング処理時間を短縮できることを確認した。また、逐次判定法に比べて独立判定法が最大通信ホップ数を小さくできるが、部分条件の数が多かたり真になる確率が小さい場合には平均通信量が多くなることが判明した。

今後、部分条件が真になる確率を考慮してトリガグラフの一部のみ逐次判定法を用いて処理時間を短縮する手法や、実システムの開発を考えている。

謝辞 本研究の一部は、文部科学省科学研究費補助金・基盤研究 (A) (26240013)、基盤研究 (B) (15H02702)、および JST 国際科学技術共同研究推進事業 (戦略的国際共同研究プログラム) の研究助成によるものである。ここに記して謝意を表す。

参考文献

- [1] Lei, C., Rundensteiner, E.A. and Guttman, J.D.: Robust Distributed Stream Processing, *Proc. International Conference on Data Engineering (ICDE2013)*, pp.817–828 (2013).
- [2] Lohrmann, B., Warneke, D. and Kao, O.: Nephel Streaming: Stream Processing under QoS Constraints at Scale, *Journal of Cluster Computing*, Vol.17, No.1, pp.61–78 (2014).
- [3] Lohrmann, B., Janacik, P. and Kao, O.: Elastic Stream Processing with Latency Guarantees, *Proc. International Conference on Distributed Computing Systems (ICDCS2015)*, pp.399–410 (2015).
- [4] Dutta, S., Narang, A. and Bera, S.K.: Streaming Quotient Filter: A Near Optimal Approximate Duplicate Detection Approach for Data Streams, *Proc. VLDB Endowment*, Vol.6, No.8, pp.589–600 (2013).
- [5] Papapetrou, O., Garofalakis, M. and Deligiannakis, A.: Sketch-based Querying of Distributed Sliding-Window Data Stream, *Proc. VLDB Endowment*, Vol.5, No.10, pp.992–1003 (2012).
- [6] ALSOK, Number of Contracts, available from (<http://www.alsok.co.jp/en/ir/finance/contractor.html>).
- [7] SECOM, 会社概要, available from (<http://www.secom.co.jp/corporate/outline/about.html>).
- [8] Polyzotis, N., Skiadopoulos, S., Vassiliadis, P., Simitsis, A. and Frantzell, N.: Meshing Streaming Updates with Persistent Data in an Active Data Warehouse, *IEEE Trans. Knowledge and Data Engineering*, Vol.20, No.7, pp.976–991 (2008).
- [9] Zhang, Q., Liu, J. and Wang, W.: Approximate Clustering on Distributed Data Streams, *Proc. International Conference on Data Engineering (ICDE2008)*, pp.1131–1139 (2008).
- [10] Stanford Stream Data Manager, available from (<http://infolab.stanford.edu/stream/>).
- [11] Open Sensor Data, available from (<http://opensensordata.net/>).
- [12] SPIRIT: Streaming Pattern Discovery, available from (<http://www.cs.cmu.edu/afs/cs/project/spirit-1/www/>).

推薦文

本論文では、IoT (Internet of Things) 環境においてセンサが接続される処理サーバがネットワーク上に分散して存在する場合において、センサデータを収集する際にストリーミング処理時間を短縮する手法を提案している。具体的には、データ収集を目的とした連続問合せを複数の部分条件に分割して処理を行うものであるが、その際に、部分条件が偽となったときには以降の処理を打ち切ることで、処理量の短縮を実現している。このような処理打ち切りを提案する本論文には高い新規性が認められる。また、提案する手法の性能評価結果からも提案手法は従来手法と比べて最大通信ホップ数と平均通信料を削減できることが確認されており、有用性も高く評価できる。よって、本論文を情報処理学会論文誌に推薦する。

(マルチメディア通信と分散処理研究会主査 重野 寛)



義久 智樹 (正会員)

2002年大阪大学工学部電子情報エネルギー工学科卒業。2003年同大学大学院情報科学研究科マルチメディア工学専攻博士前期課程を修了、2005年同専攻博士後期課程修了。博士(情報科学)。2005年京都大学学術情報メディアセンター助教。2008年大阪大学サイバーメディアセンター講師を経て2009年より同准教授となり、現在に至る。この間、カリフォルニア大学アーバイン校客員研究員。ストリーミング配信や処理に関する研究に従事。電子情報通信学会、IEEE各会員。



原 隆浩 (正会員)

1995年大阪大学工学部情報システム工学科卒業。1997年同大学大学院工学研究科博士前期課程修了。同年同大学院工学研究科博士後期課程中退後、同大学院工学研究科助手、2002年同大学院情報科学研究科助手、2004年同大学院情報科学研究科准教授。2015年より同大学院情報科学研究科教授となり、現在に至る。工学博士。1996年本学会山下記念研究賞受賞。2000年電気通信普及財団テレコムシステム技術賞受賞。2003年本学会研究開発奨励賞受賞。2008年、2009年本学会論文賞、2015年日本学術振興会賞受賞。モバイルコンピューティング、ネットワーク環境におけるデータ管理技術に関する研究に従事。IEEE、ACM、電子情報通信学会、日本データベース学会の各会員。