



データベース放送システムのためのサーバと移動型クライアントによる協調型問合せ処理方式

加下, 雅一
寺田, 努
原, 隆浩
塚本, 昌彦
西尾, 章治郎

(Citation)

情報処理学会論文誌データベース (TOD) , 44(SIG08(TOD18)):92-104

(Issue Date)

2003-06-15

(Resource Type)

journal article

(Version)

Version of Record

(URL)

<https://hdl.handle.net/20.500.14094/90002124>



データベース放送システムのための サーバと移動型クライアントによる協調型問合せ処理方式

加 下 雅 一[†] 寺 田 努^{††} 原 隆 浩^{†††}
塚 本 昌 彦^{†††} 西 尾 章 治 郎^{†††}

近年、サーバが携帯端末や PDA などの移動型クライアントにデータベースの内容を定期的に放送する放送型データベースシステムが注目されている。放送型データベースシステムにおける問合せ処理方式としては、クライアントが問合せに関係するテーブル全体を蓄積して処理を行う方式と、サーバが問合せ処理を行い、結果をクライアントに放送する方式が考えられる。前者ではクライアントのディスクサイズが小さい場合、後者では問合せが頻繁に発生する場合に、クライアントが問合せ結果を受け取ることができない。そこで本論文では、これらの問題点を解決する問合せ処理方式を提案する。提案方式では、クライアントが問合せに関係するタプルだけを蓄積できるように、放送データに識別子を付加する。また、サーバがデータの受信方法を指示するルールを作成してクライアントに送信することで、クライアント側において問合せ結果のテーブルを再現できる。本方式を用いることで、放送型データベースシステムにおいてクライアントが問合せ結果を効率的に受け取れるようになる。

A Collaborated Query Processing Method by a Server and Mobile Clients for a Database Broadcasting System

MASAKAZU KASHITA,[†] TSUTOMU TERADA,^{††} TAKAHIRO HARA,^{†††}
MASAHIKO TSUKAMOTO^{†††} and SHOJIRO NISHIO^{†††}

Recently, there has been an increasing interest in the broadcast database system where a server periodically broadcasts contents of a database to mobile clients such as portable computers and PDAs. There are two methods to process a query in the broadcast database system; one is that a client stores in his/her disk all data that are necessary in processing the query and then processes it locally, and the other is that a server processes a query and broadcasts the query result to the client. However, clients cannot properly get the query result when the disk space of the clients is small in the former method or when queries are issued frequently in the latter method. In this paper, to resolve this problem, we propose a method where a server and a client collaboratively process a query. In this method, the server adds identifiers to tuples appearing in the query result, and thus, the client can store only necessary tuples by referring to the identifiers. In addition, the server broadcasts rules which define the client's behavior for receiving data, and then, the query results are constructed on the client by using the rules. In this way, clients can get query results efficiently.

1. はじめに

近年、無線通信技術の発展にともない、放送型通信を用いて情報を配信する放送型情報システムが注目されている。放送型情報システムでは、サーバはクライアントへの広い帯域幅を利用して多種のデータを周期

的に放送し、クライアントは必要なデータのみを選択して取得する。放送型情報システムでは、クライアント数が増加してもデータ配信のコストがほとんど変わらないため、クライアント数が多い場合に通信品質を落とさず情報配信ができ、さらに、データアクセスのスループットの向上が期待できる。

これまでに、放送型情報システムの性能向上を目的とし、放送データのスケジューリング戦略^{1),4),11)~14)}、クライアント側のキャッシュ戦略¹⁾、データ更新の反映²⁾、プッシュ型とプル型の融合戦略^{3),6)}、放送を用いたプル型通信におけるアイテムのプリフェッチ戦略⁵⁾など多くの研究が行われている。

[†] 大阪大学大学院工学研究科
Graduate School of Engineering, Osaka University

^{††} 大阪大学サイバーメディアセンター
Cybermedia Center, Osaka University

^{†††} 大阪大学大学院情報科学研究科
Graduate School of Information Science and Technology, Osaka University

これらの研究では、放送データを単なるデータアイテムとして扱っており、具体的な放送内容やデータ形式に基づいてシステムの効率化を行っているものは少ない。しかし放送型情報システムでは、アプリケーションに依存してハイパーリンク形式やリレーショナルデータモデル形式など、様々なデータ形式が存在するため、放送するデータの内容や形式に則したデータ処理機構が、性能向上の重要な要因となる。そこで本論文では、サーバがリレーショナルデータモデルに基づくデータベースの内容を繰り返し放送し、ユーザが、放送されるデータベースに対して問合せを発行する環境を想定する。このようなシステムを放送型データベースシステムと呼ぶ。

本論文では、放送型データベースシステムにおいて効率的な問合せ処理を実現することを目的とする。具体的には、サーバとクライアントが協調して問合せ処理を行うことで、問合せ処理の際に必要なクライアントの記憶領域を削減し、応答時間の節約を可能にする。

以下、2章では、本研究で想定する環境について述べる。3章では放送型データベースシステムについて述べ、放送型データベースシステムにおける一般的な問合せ処理方式を説明する。4章で提案する協調型方式について説明し、5章で協調型方式の性能評価を行う。6章で本研究の考察を行い、最後に7章で本論文のまとめと今後の課題について述べる。

2. 想定環境

本研究では、街中で不特定多数のユーザに周辺情報を配信するといったアプリケーションを想定している。その一例として、ショッピングセンターにおける情報サービスがあげられる。このサービスでは、サーバがショッピングセンター内の広告情報や店舗情報、また店舗で扱っている商品情報を含むデータベースを放送し、ユーザは携帯端末を持ち歩きながら放送される情報を受信し利用する。サーバが放送しているデータベースは、店舗の地図画像や商品画像を含み、画像の数はデータベース全体で数万枚、サイズは数百メガバイトとする。クライアントは数千の規模で存在し、各クライアントは、放送されている情報を絶えず受信していることを想定する。通常、クライアントは放送を受信することのみで要求を満たしているが、「商品Aの画像とその商品を扱っている店舗の地図が欲しい」といった自然結合演算などの複雑な演算をとまう情報検索を行いたい場合には、サーバに対して問合せを発行するものとする。その頻度は数十分に1程度を想定する。そのため、サーバに問合せが到着する平均

間隔は数秒となる。ユーザは、ショッピングをしながら欲しい商品を検索するような環境を想定する。そのため、クライアントは数分程度の応答時間なら許容でき、問合せ処理にはリアルタイム性を要求しないものとする。サーバからクライアントへの放送は比較的限られた地理的範囲で行えることを想定し、放送帯域は10 Mbps程度とする。

3. 放送型データベースシステムにおける問合せ処理

本章では、まず本研究で想定する放送型データベースシステムについて述べ、次に放送型データベースシステムにおいて一般的な問合せ処理方式であるクライアント型方式とオンデマンド型方式について説明する。

3.1 放送型データベースシステム

本研究では、図1のように、放送型情報システムにおいてリレーショナルデータベースの内容を放送し、ユーザ（クライアント）が問合せを行う環境を想定する。放送型データベースシステムは次に示す特徴を持つ。

- (1) **放送する内容**：サーバは、リレーショナルデータベースの内容を周期的に放送する。
- (2) **クライアント**：放送を受信するクライアントとして、記憶領域、電力資源、処理能力の乏しい携帯端末を想定する。
- (3) **ダウンリンク**：サーバからクライアントへの放送帯域は、2つのチャンネルに分割されているものとする。図1に示すように、広帯域のメイン放送帯域を用いて、データベースの内容を繰り返し放送し、狭帯域のサブ放送帯域を用いてそれ以外のデータを放送する。
- (4) **アップリンク**：クライアントから、サーバへの狭帯域の通信チャンネルが存在する。このアップリンクを用いて、クライアントは問合せをサーバに送信する。

放送型データベースシステムにおいて、放送されるデータに問合せを行う場合、一般に次の2つの方式が考えられる。

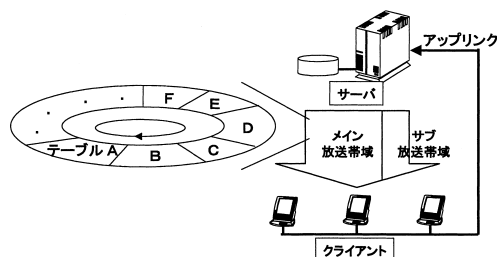


図1 放送型データベースシステム

Fig. 1 A database broadcasting system.

● **クライアント型方式**：問合せに関係するすべてのテーブルをクライアントのディスクにいったん蓄え、必要なすべてのデータが揃ってからクライアント上で問合せ処理を行う。

● **オンデマンド型方式**：クライアントがアップリンクを利用して問合せをサーバに送信し、サーバが問合せ処理を行った後でサブ放送帯域を用いて問合せ結果をクライアントに配信する。

以下では、クライアント型方式、オンデマンド型方式の詳細を説明する。

3.2 クライアント型方式

クライアント型方式では、次の手順で問合せが行われる。

- (1) クライアント上で、放送されるデータベースに対して SQL 文による問合せが発生する。
- (2) クライアントは問合せ発行後、メイン放送帯域を監視し、問合せに関係するすべてのテーブルを受信してローカルディスクに蓄積する。
- (3) クライアントは、蓄積したテーブルに対して問合せ処理を行い、問合せ結果を得る。

クライアント型方式では、クライアント上で問合せ処理が完結するため、クライアント数が増加しても、1 放送周期以内に問合せに関係する必要なすべてのデータを蓄積でき、問合せ結果を得ることができる。また、アップリンクを使用しないため、アップリンクを用意できない環境でも動作する。しかし、クライアント型方式には以下のような問題点がある。

- (1) クライアントのディスク使用量が多い。

問合せに関連するすべてのテーブルをディスクに蓄えて処理するため、クライアントのディスク容量を圧迫する。たとえば、図 2 のように、テーブル X, Y を含むデータベースが放送されている環境で、図中に示す SQL 文によって問合せを行う場合、クライアントはまず問合せ中に含まれるテーブルを蓄積してから処理を行う。この問合せを処理するのに必要なタプルは斜線の入った 4 つのタプルだけであり、さらに使用された属性は、Attri-1, Attri-3 だけである。このように、クライアント型方式では問合せ結果に直接現

れないタプルや属性のデータが蓄積され、ディスク領域が圧迫されることになる。特に、クライアントが十分なディスク領域を持っていないときは、問合せを処理できない場合がある。

- (2) クライアントにかかる負荷が大きい。

問合せ処理は一般に負荷の高い処理となるため、クライアントの処理能力が低い場合、問合せ処理に処理能力のほとんどを奪われてしまう。また、処理コストは問合せの複雑さによって大きく変化するため、必要なスペックを予想することが難しい。

3.3 オンデマンド型方式

オンデマンド型方式では、次の手順で問合せが実行される。

- (1) クライアント上で、放送されるデータベースに対する問合せが発生する。問合せはアップリンクを利用してサーバに送信される。
- (2) サーバは、問合せ処理を行い、問合せ結果をサブ放送帯域を利用して放送する。
- (3) クライアントは、自分宛に放送される問合せ結果を受信する。

本研究では、多くのクライアントはメイン放送帯域を用いて送信されてくるデータを受信することで要求を満たしており、放送をつねに受信している受信専用端末が多く存在することを想定している。そのため、問合せ結果の放送にはメイン放送帯域は用いない。

オンデマンド型方式では、問合せ処理のすべてをサーバが実行し、クライアントは放送される結果を受け取るだけでよい。ため、問合せを処理するためのディスク領域を必要としない。また、発生する問合せ数が少ない場合、問合せ結果が放送されるまでの待ち時間がなく、クライアントはすぐに結果を取得できる。しかし、オンデマンド型方式では、問合せが頻繁に起こる場合や問合せの結果サイズが大きい場合にサブ放送帯域が枯渇するため、クライアントが問合せ結果を受信するまでに長い時間がかかる可能性がある。

4. 協調型方式

3 章で述べたように、クライアント型方式とオンデマンド型方式ともに、状況によってはクライアントが問合せ結果を長時間受け取れなかったり、問合せ処理自体を行えなかったりする可能性がある。そこで、本章ではこの問題を解決するため、放送型データベースシステムにおいて効率的な問合せ処理が可能な協調型方式を提案する。まず協調型方式の概要を説明し、次にその処理手順について詳細に述べる。

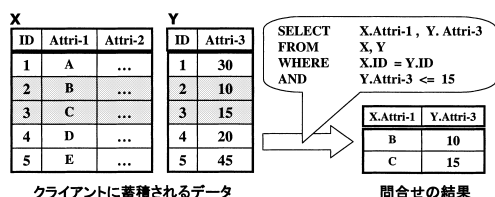


図 2 クライアント型方式における問合せ処理

Fig. 2 Query processing in the client method.

4.1 協調型方式の概要

協調型方式では、サーバとクライアントが協調して問合せ処理を行うことで、クライアント型方式に比べてクライアントのディスク使用量を小さくし、オンデマンド型方式に比べて応答時間を低減する。協調型方式の特徴を以下に示す。

サーバによる問合せ処理の補助： クライアントは、アップリンクを利用して問合せをサーバに送信する。問合せを受け取ったサーバは問合せを処理し、問合せ結果に含まれるタプルに処理用の識別子を付加する。クライアントはメイン放送帯域を用いて放送されるデータベースのうち、識別子を参照して必要なデータのみを蓄積する。そのため、クライアントのディスク使用量を低減できる。

ルールによる受信データ処理の指定： サーバは、タプルに識別子を付加した後、クライアントがデータを処理するためのルールを作成し、サブ放送帯域を用いて放送する。クライアントが自分宛に放送されたルールを受信すると、ルールが自動的に問合せ結果に含まれるタプルのみを蓄積し、問合せ結果を再現する。一般にルールのサイズは非常に小さいため、問合せ結果をそのまま放送するのに比べて放送帯域を圧迫しない。

4.2 ECA ルール

協調型方式では、クライアントのデータ受信処理方法を記述するためにルールを用いる。ルールの書式には、発生する事象（イベント）、実行させるための条件（コンディション）、イベントの発生によって実行される操作（アクション）の3つを1組としたECAルール（Event, Condition, Action）を用いる。ECAルールは、データベース技術の1つであるアクティブデータベースの動作記述言語として用いられている。アクティブデータベースとは、データベース内部で起こる事象を監視し、あらかじめ定義された条件に適応する事象の発生に反応して、自動的に更新などの操作を行うデータベースシステムである⁸⁾。協調型方式におけるクライアントの動作記述にECAルールを用いることで、次のような利点がある。

- イベント駆動型であるため、データの到着をイベントとして検出する機構を用意しておけば、到着データに対する要求をルールとして記述することで、必要な動作が自動的に実行される。問合せ処理は、このような要求の集合として表現できる。
- 処理はすべてECAルールの組で表現されるため、ECAルールを変更、追加、削除することで機能のカスタマイズが可能である。

表 1 使用できるイベント

Table 1 Events.

名称	内容
SELECT	テーブルに対するデータ参照
INSERT	テーブルに対するタプル挿入
DELETE	テーブルのタプル削除
UPDATE	テーブルのタプル更新
RECEIVE	データの到着
TIMER	設定したタイマの発火

表 2 使用できるアクション

Table 2 Actions.

名称	内容
QUERY([クエリー内容])	データベース操作
ENABLE_ECA([ルール識別子])	ECA ルールの有効化
DISABLE_ECA([ルール識別子])	ECA ルールの無効化
INSERT_ECA([ルール内容])	ECA ルールの格納
DELETE_ECA([ルール識別子])	ECA ルールの削除
SET_TIMER([タイマ識別条件],[時間])	新たなタイマの設定
KILL_TIMER([タイマ識別子])	タイマの削除
STORE([QueryID],[テーブル名], [テーブル名],[属性],...)	タプルの格納
MATCH([QueryID],[テーブル名], [比較対象テーブル名],[属性],...)	格納済みのタプルと 組み合わせで格納
DISPLAY([QueryID])	問合せ結果の表示

表 3 NEW データと OLD データの内容

Table 3 NEW data and OLD data.

イベント	NEW	OLD
SELECT	参照タプル	-
INSERT	挿入タプル	-
DELETE	-	削除タプル
UPDATE	更新後タプル	更新前タプル
RECEIVE	到着パケット内容	-
TIMER	タイマ識別子	-

本論文で用いるECAルールで使用可能なイベントおよびアクションを表1、表2に示す。また、到着したデータの内容に基づいて処理を行うルールでは、イベント対象となったタプル情報などが必要になるため、NEW データ、OLD データと呼ぶシステム変数を用意する。イベント発生時にこれらの変数に必要な情報が自動的に格納され、ルール中で使用できる。各イベントに対するNEW データ、OLD データの内容を表3に示す。

4.3 問合せ処理アルゴリズム

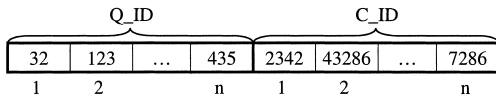
提案方式の問合せ処理手順は以下のとおりである。

(1) 問合せの発生と送信

クライアント上で問合せが発生し、アップリンクを利用してサーバに送信される。

(2) 識別子の書き込み

サーバは、受け取った問合せに対して一意の識別子（QueryID）を割り当てる。放送するデータベースの

図3 Q_ID , C_ID の例Fig. 3 An example of Q_ID and C_ID .

各タプルには、処理用の識別子を書き込むための属性として、問合せ識別子 Q_ID 、組合せ識別子 C_ID を用意しておく。あるタプル中の各属性のデータ例を図3に示す。各属性は複数の領域に分割されており、その領域に識別子を書き込む。各属性に用意する領域の数はあらかじめ決定されており、本研究ではこの領域数を最大識別子数と呼ぶ。図3では、これを n としている。サーバは受信した問合せを解析し、問合せ結果に含まれるタプルを調べ、そのタプルの問合せ識別子 Q_ID の空き領域に *QueryID* を書き込む。クライアントは送信されてくるタプルの属性 Q_ID を参照し、特定の識別子を含むタプルのみを蓄積できるため、クライアント型方式に比べ受信するデータサイズを低減できる。また、結合演算のように複数のテーブルから問合せ結果が得られる場合、問合せ結果を構成する複数のタプルに対して、組合せ識別子 C_ID の領域に同じ値を書き込む。クライアントは、組合せ識別子 C_ID が等しいタプルを結合することで問合せ結果を再現できる。たとえば、問合せ結果を構成する2つのタプルにおいて、 Q_ID の k_1 , k_2 番目の領域に該当する *QueryID* が書き込まれたものとする、両タプルの C_ID の k_1 , k_2 番目の領域の値を比較し、それが一致するものを結合する。

(3) ECA ルール作成

サーバは、クライアントが問合せ結果に含まれるタプルのみを受信し、結果を再現するためのECAルールを作成する。サーバは、問合せに関係するテーブル数や集約関数の有無など、問合せの種類ごとに、複数のECAルールをセットにしたテンプレートを用意しておく。サーバは問合せを解析し、どの種類のテンプレートを適用すべきかを決定した後、必要な値を選択されたテンプレートに記入することでECAルールを作成する。またサーバは、問合せ結果に含まれるタプルが放送される時間を調べ、その時間だけ放送帯域を監視するようなECAルールを作成する。これにより、クライアントは必要なデータが放送される時間だけ放送を監視できるため、消費電力を低減できる。

(4) ECA ルール放送・受信

サーバは、作成したECAルールをサブ放送帯域を

```
SELECT X.Attri-1, Y.Attri-2
FROM X, Y
WHERE X.ID = Y.ID
AND Y.Attri-2 <= 15
```

図4 問合せ例

Fig. 4 A query example.

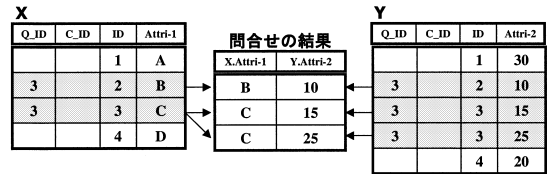


図5 問合せ識別子の書き込み

Fig. 5 Addition of query identifiers.

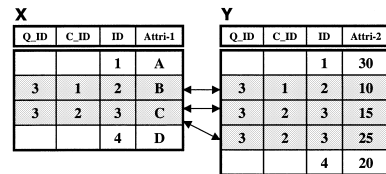


図6 組合せ識別子の書き込み

Fig. 6 Addition of combination identifiers.

用いて放送する。問合せを行ったクライアントは、自分宛のECAルールが放送されるまでつねにサブ放送帯域を監視し、それを受信する。

(5) 問合せ結果の獲得

クライアントは受信したECAルールにより、メイン放送帯域から自動的に必要なタプルを受信し、問合せの結果を再現する。また、クライアントがデータを受信し損ねた場合には、再び新たな問合せを発行させる。

(6) 問合せ識別子の解放

サーバは、クライアントがECAルールを受信し終わるまでの時間を予測し、その時間以降は、クライアントが受信済みと判断して、付加した識別子をタプルから削除する。

4.4 問合せ実行例

クライアントが、図4に示すSQL文によって問合せを行う場合の提案方式の実行例を示す。

クライアントは、問合せをアップリンクを用いて送信する。サーバは、受信した問合せに対して *QueryID* を決定する。ここでは仮に3とする。次に、図5に示すように、問合せの結果に含まれるタプルを調べ、それぞれのタプルの問合せ識別子に *QueryID* を書き込む。図中では、問合せ識別子を Q_ID 、組合せ識別子を C_ID と表す。実際には Q_ID および C_ID は複数の領域に分割されているが、簡単化のため、図中では1つの領域として表現している。そして、図6

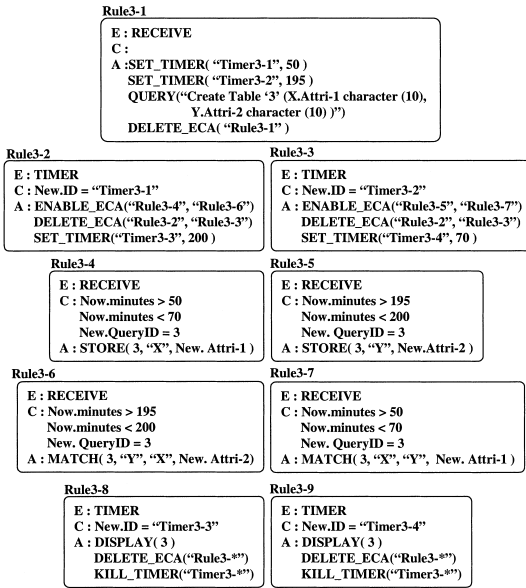


図7 作成される ECA ルール
Fig. 7 Generated ECA rules.

に示すように、結果テーブルの各タプルが元のテーブルのどのタプルの組によって構成されるのかを調べ、該当するタプルの組合せ識別子に同じ値を書き込む。

次に ECA ルールを作成する。この例では、図7に示すルール群が作成される。‘Rule3-1’は、問合せ結果のテーブルを作成するためのルール、‘Rule3-2’、‘Rule3-3’は、放送データ受信と問合せ結果表示のタイミングを計るためのルール、‘Rule3-4’、‘Rule3-5’、‘Rule3-6’、‘Rule3-7’は、データを受信し蓄積するためのルールである。‘Rule3-8’、‘Rule3-9’は結果を表示し、その後すべてのルールおよびタイマを削除するルールである。クライアントがタプルを受信する場合、テーブル X とテーブル Y の放送順序によって、結合の順序や結果を表示するタイミングが異なる。そこで、テーブル X が最初に放送されてきた場合には、‘Rule3-4’、‘Rule3-6’、‘Rule3-8’が使用されるように、‘Rule3-2’でタイマをセットする。また、テーブル Y が最初に放送されてきた場合には、‘Rule3-5’、‘Rule3-7’、‘Rule3-9’が使用されるように、‘Rule3-3’でタイマをセットする。ルール中の Now は現在時刻を表し、‘Rule3-4’では識別子が付加されたタプルが放送される時間である 50 秒から 70 秒の間にタプルを受信した場合、そのタプルを蓄積する。これにより、クライアントは必要なタプルが放送される時間のみ、帯域を監視することができる。

本研究では、サーバとクライアントの時刻の同期を

行うために、メイン放送帯域を用いて時刻印を放送する。クライアントは、放送を監視し時刻印を受信することで時刻を合わせることができ、サーバとクライアントの時刻のずれは、放送帯域でのわずかな伝搬遅延のみとなる。

5. 評価

本章では、次に示す3つの評価基準を用いて、クライアント型、オンデマンド型と協調型方式を比較し、協調型方式の有効性を検証する。

(1) ディスク使用量

問合せを実行するうえで、クライアントが必要とする作業領域（ディスク使用量）を比較する。ディスク使用量は、問合せ結果のサイズを含む。

(2) 応答時間

問合せ発生から問合せ結果を受け取るまでの時間を比較する。

(3) チューニング時間

問合せ発生から問合せの結果を受け取るまでに放送を監視した時間を比較する。

5.1 データベーススキーマと問合せモデル

本論文では、2章で述べたショッピングセンターにおける情報サービスをアプリケーション例として想定し、評価で用いるデータベースのスキーマと問合せモデルを決定した。データベーススキーマは、飲食店や衣料店などの各ジャンルごとに店舗テーブル {店舗ID, 店名, 画像, ...}, 商品テーブル {商品ID, 店舗ID, 商品名, 画像, ...} を持つとした。店舗テーブルは‘店舗ID’を主キーとし、店舗の名前や地図画像を属性として持ち、商品テーブルは、‘商品ID’を主キーとし、その商品が販売されている店の識別子と商品画像を属性として持つ。

店舗テーブルと商品テーブルのタプルサイズは等しいものとし、問合せは SQL によって記述されるものとする。また簡単化のため、ユーザは店舗テーブルと商品テーブルを自然結合する問合せのみを行うものとする。ただし、自然結合を行う店舗テーブルと商品テーブルは同一ジャンル内であるとする。一般的な問合せには射影演算が含まれることが多いが、本評価では簡単化のため、射影演算は含まれないものとする。本研究で想定するデータベーススキーマでは、タプルサイズのほとんどを1つの画像が占めており、射影演算が含まれたとしても、問合せ結果のサイズはほとんど変わらないため、オンデマンド型方式の平均応答時間が劇的に小さくなることはない。

簡単な問合せ例を以下に示す。

表 4 評価のパラメータ
Table 4 Parameters.

変数名	意味	値
g	店舗のジャンル数 [ジャンル]	10
n	ジャンル内ショップ数 [店/ジャンル]	40
t	ショップ数内商品数 [品/店]	100
s	1 タップルのサイズ [バイト]	5,000
i	1 つのタップルに付加可能な最大識別子数 [個]	200
b_u	アップリンク帯域 [kbps]	56
b_m	メイン放送帯域 [Mbps]	10
b_s	サブ放送帯域 [Mbps]	1
s_q	問合せのサイズ [バイト]	100
d	問合せ到着間隔の平均 [秒]	0.1~10
r_a	タップル利用率の平均	1
r_s^2	タップル利用率の分散	1
a	識別子領域のサイズ [バイト]	2
s_e	1 つの ECA ルールのサイズ [バイト]	140

```
SELECT  [店舗テーブル].*, [商品テーブル].*
FROM    [店舗テーブル], [商品テーブル]
WHERE   .....
```

5.2 評価のパラメータ

表 4 に、評価で用いるパラメータとその値を示す。パラメータの値は、実在するショッピングセンターの規模を考慮し設定した。サーバに問合せが到着する間隔を表す問合せ到着間隔はポアソン分布で与え、その平均をパラメータ d で表している。また、タップル利用率は正規分布で与え、その平均と分散を r_a , r_s^2 で表している。タップル利用率は、テーブル内の全タップルに対する問合せの結果に含まれるタップルの割合を表す。

本評価では、クライアントが発生する複数の問合せはそれぞれ独立しており、クライアントは問合せ結果を受け取る前に、次の問合せを行うことができると想定しているため、サーバに問合せが到着する間隔は応答時間に影響を受けない。また、クライアント数は数千程度と想定しているため、各クライアントが逐次的に問合せを行う場合にも、クライアントの応答時間のシステム全体への影響は無視できるほど小さく、サーバに問合せが到着する間隔に影響を与えない。

5.3 ディスク使用量の比較

クライアント型、オンデマンド型、協調型方式のディスク使用量 D_{cli} , D_{on} , D_{col} は、付録に示す式 (2), 式 (3), 式 (4) で表せる。これらの式においてタップル利用率を変化させ、3 方式のディスク使用量を比較した結果を図 8 に示す。オンデマンド型方式では、タップル利用率に比例して問合せ結果のサイズが大きくなるため、ディスク使用量は右上りの直線になっている。クライアント型方式のディスク使用量は、問合せ結果のサイズと問合せに係るテーブルのサイズの和で

表されるため、オンデマンド型方式のディスク使用量に比べ、一定量大きくなっている。一方、協調型方式では、問合せ結果に現れるタップルのみを蓄積できるため、クライアント型方式に比べディスク使用量は小さくなっている。一般的な環境でのタップル利用率がたかだか 1% 程度と仮定した場合、協調型方式のディスク使用量は約 600 K バイトとなる。クライアントが、PDA のように 10 M バイトほどしか記憶領域を持たない端末の場合、クライアント型方式では、問合せ処理に必要なすべてのデータを蓄積できない場合があるが、協調型方式では十分に蓄積できる。

5.4 応答時間の比較

クライアント型、オンデマンド型、協調型方式における応答時間 R_{cli} , R_{on} , R_{col} は、表 4 のパラメータを用いると付録に示す式 (7), 式 (10), 式 (19) で表される。これらの式において問合せ発生間隔を変化させ、応答時間を比較した結果を図 9 に示す。クライアント型方式では、クライアント自身が問合せ処理に必要なデータを蓄積し処理するため、問合せ発生間隔は応答時間に影響しない。オンデマンド型方式では、問合せ発生間隔が 3.4 秒以下では、グラフが切れている。これは、サーバのサブ放送帯域の放送キューにおいて、問合せの到着間隔が放送データがキューから削除されるまでの時間に比べ小さくなり、待ち行列が破綻してしまうためである。破綻した待ち行列における応答時間は非常に大きくなる。協調型方式はクライアント型方式に比べ、わずかに応答時間が長くなっている。これは、各テーブルが識別子保持用の領域分だけ大きくなり、放送周期が長くなるためである。協調型方式では、問合せ発生間隔が 0.6 秒以下の領域では、グラフが切れている。これは、サーバの識別子を確保するためのキューにおいて、問合せの到着間隔が識別子領域が開放されるまでの時間に比べ小さくなり、待ち行列が破綻してしまうためである。破綻した待ち行列における応答時間は非常に大きくなる。問合せ発生間隔が小さい場合、オンデマンド型方式に比べ協調型方式の応答時間は小さいため、協調型方式の方が有効である。しかし、問合せ発生間隔が大きい場合、オンデマンド型方式の方が応答時間が短くなる。

タップル利用率 r を 0.5% および 2% に設定した場合の応答時間を図 10 に示す。タップル利用率が大きくなるにつれ、問合せ結果のサイズが大きくなりサブ放送帯域が枯渇しやすくなるため、グラフが右上に移動する。

次に、協調型方式における最大識別子数 i を 100 および 400 に設定した場合の応答時間を図 11 に示す。

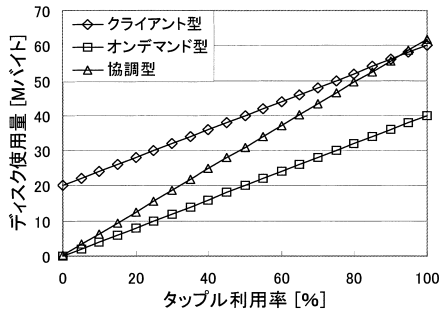


図 8 データ利用率とディスク使用量

Fig. 8 The tuple usage rate vs. the disk space consumption.

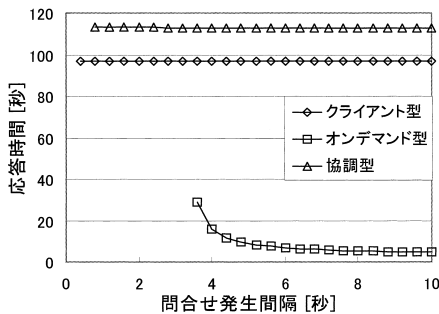


図 9 問合せ発生間隔と応答時間

Fig. 9 The tuple usage rate vs. the response time.

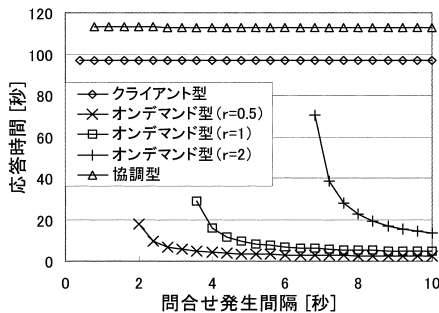


図 10 タプル利用率と応答時間

Fig. 10 The bandwidth of sub-channel vs. the response time.

グラフ中では、ある問合せ発生間隔より小さい領域では、識別子領域が開放されるまでの時間より、問合せが到着する間隔が小さくなるため、識別子を確保するための待ち行列が破綻してしまう。そのため、解析的に応答時間を求めることはできない。最大識別子数が大きくなると、各タプルに付加される識別子のサイズが大きくなるため応答時間は長くなるが、識別子のための領域が枯渇する可能性が低くなるため、グラフは左上に移動する。

最後に、サブ放送帯域 b_s を 2 Mbps および 3 Mbps に設定した場合の応答時間を図 12 に示す。グラフ中

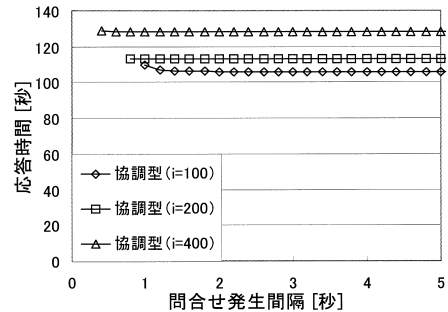


図 11 最大識別子数と応答時間

Fig. 11 The interval of query generation vs. the response time.

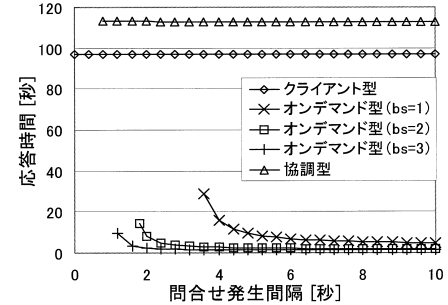


図 12 サブ放送帯域と応答時間

Fig. 12 The max number of identifiers vs. the response time.

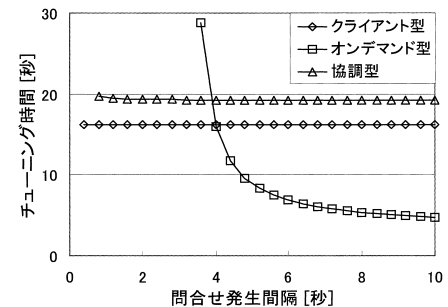


図 13 問合せ発生間隔とチューニング時間

Fig. 13 The interval of query generation vs. the tuning time.

では、ある問合せ発生間隔より小さい領域では、1つの問合せ結果を放送する時間より、問合せが到着する間隔が小さくなるため、サブ放送帯域の放送キューが破綻してしまう。サブ放送帯域が大きくなれば、オンデマンド型方式ではデータを短時間で送信できるため応答時間が短くなる。一方、サブ放送帯域が枯渇しにくいため、放送キューが破綻してしまう問合せ発生間隔は小さくなる。

5.5 チューニング時間の比較

クライアント型、オンデマンド型、協調型方式におけるチューニング時間 T_{cli} , T_{on} , T_{col} は、表 4 のパラ

メータを用いると付録に示す式 (9), 式 (18), 式 (30) で表される. これらの式において問合せ発生間隔を変化させ, チューニング時間を比較した結果を図 13 に示す. クライアント型方式, 協調型方式のチューニング時間は, 図 9 に示す応答時間と比べて小さくなる. これは, クライアント型方式ではインデックス情報を用いることで必要なテーブルが放送される時間だけ放送を監視できるためである. 協調型方式では ECA ルール中に必要とする情報が放送される時間が記述されており, その時間だけ放送を監視できるためである. 一方, オンデマンド型方式では, 問合せを送信したクライアントは, 問合せ結果が放送されるまでサブ放送帯域を監視しなければならないため, チューニング時間は応答時間と等しくなる.

6. 考 察

提案方式を実環境で運用する際には, いくつかの技術課題がある. そこで本章では, まずこれらの技術課題について考察する. さらに, 本研究と類似したアプローチを持つ従来研究について述べ, それらとの比較を行う.

6.1 識別子の付加方法

協調型方式において, 問合せごとにデータベース中の各タプルに対して識別子を付加および削除する作業は, サーバの負荷が大きい. この負荷の軽減は今後の課題の 1 つであるが, 付加する識別子と問合せの情報をデータベースとは別に保持しておき, 各タプルを放送する際にそれに付随する識別子を放送するといった方法が考えられる.

また, 協調型方式において 1 つのタプルに付加できる識別子の数を大きくしすぎると, メイン放送帯域で放送するデータの放送周期が長くなり, 応答時間が長くなる. 逆に, 小さくしすぎても同時に処理できる問合せ数が少なくなり, 応答時間が長くなる. したがって, 協調型方式を用いる場合には, 適切な識別子領域のサイズを決定する必要がある. これについては今後の課題であるが, システム環境に応じて固定値を割り当てる手法や, 1 周期ごとに適応的に決定する手法などを検討している.

6.2 ECA ルールの作成と実行

協調型方式では, 問合せの種類ごとにテンプレートが用意されており, そのテンプレートに必要な値を代入することで ECA ルールを生成している. これにより, サーバにおける ECA ルール作成の負荷を軽減しているが, テンプレートで対処しきれない複雑な問合せの場合はこの方式を適用できない. 現在は, 1 つの

テーブルを用いた射影, 選択演算と 2 つのテーブル間の自然結合演算のみに対応しており, 積演算や集約関数を含む演算を処理することはできない. しかし, ECA ルールのアクションを追加し, 識別子領域の使用方法を改良した後, テンプレートの種類を充実させることで, 複雑な問合せに対応できるものとする. たとえば, 集約関数を含む問合せを処理する場合, いったんクライアント端末上に必要なタプルを蓄積し, その後, 集約演算を行うアクションを実行するような ECA ルールを作成することで実現できる.

また, 本論文では, テンプレートに値を書き込むことで簡単に ECA ルールを作成することを重視しているため, 各問合せに対して作成されるルールのは数は, 参照するテーブル数を N とすると $O(N^2)$ になる. しかし, 比較的単純な問合せを想定しているため, これは大きな問題にはならないと考える. 一方, 多くのテーブルを使用する複雑な問合せに対しても, より少ないルール数で効率的に処理を記述できる ECA ルールを作成することは可能である.

ECA ルールの実行には, イベントの監視やコンディションの比較などの処理が必要となる. 本論文で例として取り上げたような単純な問合せでは, クライアントが ECA ルールを処理する負荷と, 問合せをそのまま処理する負荷にそれほど違いはないが, 問合せが複雑になるにつれて後者の方が大きくなる. したがって提案方式は, クライアントの負荷を大幅に軽減できるものと考えられる.

6.3 様々な環境における有効性

5 章で述べたように, アプリケーションによりデータベースサイズやスキーマ設計などのパラメータは強く影響をうける. 本論文では 1 つの代表的な状況を想定した評価について示したが, その他の環境での提案方式の有効性については論じていない. しかし, いくつかの異なる環境を想定して同様の評価を行っており, その結果, 3 手法の相対的な性能はほとんど変化しないことを確認している.

6.4 関連研究

これまでに, 放送型情報システムの性能向上を目的として多くの研究が行われている. 代表的なものに放送ディスク¹⁾がある. 放送ディスクでは, クライアントのデータアイテムに対するアクセス確率に偏りがあることに注目し, よくアクセスされるアイテムを頻繁に放送することで応答時間を低減している. さらに文献 9) では, 放送ディスクにおける放送スケジュール作成に木構造の概念を導入することで, 計算コストを削減している.

これらの研究では、クライアントは自分が要求するデータアイテムをあらかじめ知っていることを仮定しているため、データベースに対する問合せのようなアイテム名を直接指定しない要求を処理できない。本研究では、ユーザの要求をデータベースに対する問合せと仮定しているため、様々なアイテム要求に対して柔軟に対応できる。また、データベースのテーブルを1つのアイテムと考えれば、頻繁にアクセスされるテーブルの放送頻度を高くするといったように、放送ディスクなど従来研究のアイデアを本研究に導入できる。

文献 3), 7) では本研究と同様に、プッシュ型とプル型の通信を融合したアプローチをとっている。文献 3) では放送チャンネルを分割し、プッシュ型とプル型通信チャンネルの最適な帯域分割について論じている。また、文献 7) ではプッシュ型とプル型の融合だけでなく、キャッシュ戦略も統合した放送スケジューリングを行っている。これらの研究では、プル型の通信チャンネルを用いてデータアイテムを放送しているのに対し、本研究では、問合せ結果や ECA ルールを放送することで、より柔軟かつ効果的にプル型チャンネルを利用している点で異なる。

本研究と同様に、放送型システムにおいてアクティブデータベースの概念を導入している研究に、SADB (Super Active Database System)¹⁰⁾がある。SADB では、放送されるデータからクライアントが必要な情報を効率的に抽出、格納、再利用するために、アクティブデータベースを用いている。SADB では、アプリケーションのすべての機能を ECA ルールで表現しているが、問合せ処理における放送帯域の利用の効率化については考慮していない。本研究では、問合せ処理における放送帯域の効率的な利用のために、アクティブデータベースを用いている点で SADB と異なる。

7. おわりに

本論文では、放送型データベースシステムにおける効率的な問合せ処理方式として協調型方式を提案した。協調型方式では、サーバが問合せ結果に含まれるタプルに識別子を付加し、データの受信方法を ECA ルールによってクライアントに指示することで、クライアントが問合せ結果を得るために必要なタプルだけを蓄積できる。性能評価の結果から、提案方式を用いることでクライアント型方式に比べてクライアントのディスク使用量を低減でき、オンデマンド型方式に比べて、問合せが頻繁に発生する場合の応答時間を低減できることを確認した。

今後の課題として、提案方式を実装して実測評価を

行うことで、提案方式の実環境における有効性を検証する必要がある。また、問合せ発生間隔や問合せ結果のサイズなどのパラメータの変化に応じて、クライアント型方式、協調型方式、オンデマンド型方式を適応的に使い分ける自動選択方式についても検討を行う。3 方式を適応的に選択し、その時点で最良の方式を選択することで、問合せをより効率的に処理できるものと考えられる。

謝辞 本研究は、文部科学省 21 世紀 COE プログラム (研究拠点形成費補助金)、文部科学省科学技術振興調整費「モバイル環境向 P2P 型情報共有基盤の確立」、文部科学省科学技術振興調整費若手任期付研究者支援「フィルタリングの数学的基盤の確立」、日本学術振興会若手研究 (B) (13780331)、および文部科学省特定領域 (14019063) 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) Acharya, S., Franklin, M. and Zdonik, S.: Broadcast Disks: Data Management for Asymmetric Communication Environments, *Proc. ACM SIGMOD*, pp.199–210 (1995).
- 2) Acharya, S., Franklin, M. and Zdonik, S.: Disseminating Updates on Broadcast Disks, *Proc. VLDB Conference*, pp.354–365 (1996).
- 3) Acharya, S., Franklin, M. and Zdonik, S.: Balancing Push and Pull for Data Broadcast, *Proc. ACM SIGMOD*, pp.183–194 (1997).
- 4) Aksoy, D. and Franklin, M.: Scheduling for Large-Scale On-Demand Data Broadcasting, *Proc. IEEE INFOCOM*, pp.651–659 (1998).
- 5) Aksoy, D., Franklin, M. and Zdonik, S.: Data Staging for On-Demand Broadcast, *Proc. VLDB Conference*, pp.571–580 (2001).
- 6) 箱守 聰, 田辺雅則, 石川裕治, 井上 潮: 放送型通信/オンデマンド型通信を統合した情報提供システム, 情報処理学会研究報告, Vol.34, No.8, pp.55–60 (1997).
- 7) Hu, Q., Lee, D. and Lee, W.: Performance Evaluation of a Wireless Hierarchical Data Dissemination System, *Proc. 5th Annual International Conference on Mobile Computing and Networking*, pp.163–173 (1999).
- 8) Lohman, G., Bruce, L., Hamin, P. and Bernhard, K.: Extensions to Starburst: Object, Types, Functions, and Rules, *Comm. ACM*, Vol.34, No.10, pp.94–109 (1991).
- 9) Peng, W. and Chen, M.: Dynamic Generation of Data Broadcasting Programs for a Broadcast Disk Array in a Mobile Computing Environment, *Proc. Int'l Conf. on Information*

Knowledge Management (ACM CIKM 2000), pp.38–45 (2000).

- 10) 寺田 努, 塚本昌彦, 西尾章治郎: 放送型データ受信のためのアクティブデータベースシステムの設計と実装, 電子情報通信学会論文誌, Vol.J83-D-I, No.12, pp.1272–1283 (2000).
- 11) Yajima, E., Hara, T., Tsukamoto, M. and Nishio, S.: Interval Optimization of Correlated Data Items in Data Broadcasting, *Proc. Int'l Conf. on Advances in Information Systems (ADVIS 2000)*, pp.127–136 (2000).
- 12) Yajima, E., Hara, T., Tsukamoto, M. and Nishio, S.: Scheduling Strategies of Correlated Data in Push-Based Systems, *Information Systems and Operational Research (INFOR)*, pp.152–173 (2001).
- 13) Yajima, E., Hara, T., Tsukamoto, M. and Nishio, S.: Scheduling and Cashing Strategies for Broadcasting Correlated Data, *Proc. ACM Symposium on Applied Computing (ACM SAC 2001)*, pp.504–510 (2001).
- 14) Hu, Q., Lee, D. and Lee, W.: Performance evaluation of a wireless hierarchical data dissemination system, *Proc. MobiCom'99*, pp.163–173 (1999).

付 録

表 4 に示すパラメータを用いて, 各方式のディスク使用量, 応答時間およびチューニング時間を定式化する.

A.1 ディスク使用量

クライアント型方式では, クライアントは問合せに関係するすべてのテーブルをディスクに蓄積する必要があるため, ディスク使用量 D_{cli} は問合せ結果サイズと問合せに関係するテーブルの総サイズの和となり, 式 (2) のようになる. 問合せ結果のサイズは, 2 テーブルを自然結合した結果のサイズとタプル利用率の積で表される.

$$D_{cli} = \frac{2s \cdot n \cdot t \cdot r}{100} + s \cdot n \cdot (t + 1) \quad (1)$$

$$= \frac{s \cdot n \cdot t \cdot r}{50} + s \cdot n \cdot (t + 1) \quad (2)$$

オンデマンド型方式におけるディスク使用量 D_{on} は, 問合せ結果のサイズと等しくなり, 式 (3) のように表される.

$$D_{on} = \frac{s \cdot n \cdot t \cdot r}{50} \quad (3)$$

協調型方式のディスク使用量 D_{col} は, 店舗テーブルのサイズ, 商品テーブルのサイズとタプル利用率の積, 問合せ結果サイズの和で表される. 問合せの結

合属性に依存して, 店舗テーブルと商品テーブルにおいて問合せ結果に含まれるタプルの割合は変化するが, 簡単化のため, 店舗テーブルのタプルはすべて問合せ結果に含まれ, 商品テーブルは $r[\%]$ だけ含まれるものとした. そのため, ディスク使用量 D_{col} は, 式 (4) のようになる.

$$D_{col} = \frac{s \cdot n \cdot t \cdot r}{50} + n \cdot \left(\frac{t \cdot r}{100} + 1 \right) (s + 2a \cdot i) \quad (4)$$

A.2 応答時間とチューニング時間

A.2.1 クライアント型方式

クライアント型方式における放送周期 C_{cli} は, データベースサイズをメイン放送帯域で割ることで求められる.

$$C_{cli} = \frac{8(s \cdot n + s \cdot n \cdot t) \cdot g}{b_m} \quad (5)$$

$$= \frac{8s \cdot n \cdot (t + 1) \cdot g}{b_m} \quad (6)$$

よって, クライアント型方式における応答時間 R_{cli} は式 (7) で表される. 関数 F は, 必要とするテーブルが放送されるまでの放送周期に対する相対的な割合を示している. 関数 F は, ジャンル数 g とタプル数 t を引数とする関数で表される.

$$R_{cli} = C_{cli} \cdot F(g, t) \quad (7)$$

$$F(g, t) = \frac{1}{g} \cdot \left(\frac{g^2 - 1}{2g} + \frac{t^2 + 1}{2g \cdot (t + 1)^2} + 1 \right) \quad (8)$$

クライアントは, インデックス情報により, 必要とするテーブルが放送される時間をあらかじめ知っているため, テーブルを受信する間だけ放送を受信できる. そのため, クライアント型方式におけるチューニング時間 T_{cli} は, 式 (9) で表される.

$$T_{cli} = \frac{C_{cli}}{g} \quad (9)$$

A.2.2 オンデマンド型方式

オンデマンド型方式における応答時間 R_{on} は, アップリンクを用いて問合せをサーバに送信する時間 U , 問合せの処理時間, サブ放送帯域のキューにおける待ち時間 Q , 問合せ結果の放送にかかる時間 D の和で表される. ここで, 問合せの処理時間は他の項に比べ十分に小さいとすると, 応答時間 R_{on} は以下の式で表される.

$$R_{on} = U + Q + D \quad (10)$$

問合せ送信遅延 U は, 問合せサイズをアップリンクの帯域で割ることで導出できるため, 以下の式で表される.

$$U = \frac{8s_q}{b_u} \quad (11)$$

キューにおける待ち時間は、放送キューに待ち行列理論を導入することで求められる。問合せが平均 d のポアソン分布に従う間隔で到着し、各問合せ結果のサービス時間の平均 h と分散 v^2 が与えられている場合、そのキューは $M/G/1$ の待ち行列と見ることができ、 $M/G/1$ の待ち行列では、平均待ち時間 Q は、以下のように表される。

$$Q = \frac{(h^2 + v^2)}{2(d - h)} \quad (12)$$

各問合せ結果のサービス時間 S は、問合せ結果のサイズを放送帯域で割ることで求められ、以下のように表される。

$$S = \frac{8 \left(s \cdot n \cdot t \cdot 2 \cdot \frac{r}{100} \right)}{b_s} \quad (13)$$

$$= \frac{4s \cdot n \cdot t \cdot r}{25b_s} \quad (14)$$

ここで、タプル利用率 r は正規分布に従い、その平均 r_a と分散 r_s^2 が与えられていることから、サービス時間の平均 h と分散 v^2 は以下の式で与えられる。

$$h = \frac{4s \cdot n \cdot t \cdot r_a}{25b_s} \quad (15)$$

$$v^2 = \left(\frac{4s \cdot n \cdot t}{25b_s} \right)^2 r_s^2 \quad (16)$$

ただし、待ち行列が破綻しないためには、サービス時間の平均が到着間隔の平均よりも小さくなければならないため、 $h < d$ という条件を満たす必要がある。また、問合せ結果の放送にかかる時間 D は平均サービス時間と等しいため、以下の式で表される。

$$D = h \quad (17)$$

オンデマンド型方式のチューニング時間 T_{on} は応答時間 R_{on} と等しくなり、以下の式で表される。

$$T_{on} = R_{on} \quad (18)$$

A.2.3 協調型方式

協調型方式における応答時間 R_{col} は、アップリンクを用いて問合せをサーバに送信する時間 U 、識別子領域を確保するまでの待ち時間 I 、サーバでの識別子の付加および ECA ルール作成にかかる時間、サブ放送帯域のキューにおける待ち時間 Q 、ECA ルールの放送にかかる時間 D 、メイン放送帯域からタプルを受信するまでの時間 M の和で表される。ここで、サーバ側におけるデータ処理時間は他の項に比べて十分に小さいとし、無視することとする。また、表 4 で想定する環境では放送キューにおける待ち時間 Q より、識別子領域を確保するまでの待ち時間 I の方が十分

に大きくなると考えられる。つまり、サブ放送帯域が混雑する可能性よりも識別子領域が不足する可能性が高いと考えられる。そのため Q は十分に小さくするとし、 Q を無視することにする。応答時間 R_{col} は以下の式で表される。

$$R_{col} = U + I + D + M \quad (19)$$

問合せ送信遅延 U は、以下の式で表される。

$$U = \frac{8s_e}{b_u} \quad (20)$$

識別子領域を確保するまでの待ち時間は、行列理論を導入することで求められる。待ち行列における出線数を表す識別子領域が i で与えられ、問合せが平均 d のポアソン分布に従う間隔で到着し、各問合せ結果のサービス時間の平均 h と分散 v^2 が与えられている場合、そのキューは $M/G/S$ の待ち行列と見ることができる。しかし、 $M/G/S$ の待ち行列では一般解が求められていない。そのため、サービス時間を出線数を表す最大識別子数 i で割ることで新たなサービス時間を定義し、近似的に $M/G/1$ の待ち行列と見なすことにする。 $M/G/1$ の待ち行列では、平均待ち時間 I は、以下のように表される。

$$I = \frac{(h^2 + v^2)}{2(d - h)} \quad (21)$$

各問合せが識別子領域を占有する時間、つまりサービス時間は、0 以上、放送周期 C_{cli} 以下の値域における一様分布に従うとすると、サービス時間の平均 h と分散 v^2 は以下の式で与えられる。

$$h = \frac{C_{col}}{2i} \quad (22)$$

$$v^2 = \frac{C_{col}^2}{12i^2} \quad (23)$$

協調型方式における放送周期 C_{col} は、以下に示すようにデータベースサイズをメイン放送帯域で割ることで求められる。データベースサイズには、識別子領域のサイズが含まれている。

$$C_{col} = \frac{8(s \cdot n + s \cdot n \cdot t + 8n \cdot (t + 1)) \cdot g}{b_m} \quad (24)$$

$$= \frac{8n \cdot (t + 1) \cdot (s + 2a \cdot i) \cdot g}{b_m} \quad (25)$$

ただし、待ち行列が破綻しないためには、サービス時間の平均が到着間隔の平均よりも小さくなければならないため、 $h < d$ という条件を満たす必要がある。また、問合せ結果の放送にかかる時間 D は、ECA ルールの総サイズをサブ放送帯域で割ることで求められるため、以下の式で表される。ただし、2つのテーブル

間の自然結合には、9つのECAルールが作成される。

$$D = \frac{8(s_e \times 9)}{b_s} \quad (26)$$

$$= \frac{72s_e}{b_s} \quad (27)$$

メイン放送帯域からタプルを受信するまでの時間 M は以下のように表される。関数 F は、必要とするテーブルが放送されるまでの放送周期に対する相対的な割合を示している。

$$M = C_{col} \cdot F(g, t) \quad (28)$$

$$F(g, t) = \frac{1}{g} \cdot \left(\frac{g^2 - 1}{2g} + \frac{t^2 + 1}{2g \cdot (t + 1)^2} + 1 \right) \quad (29)$$

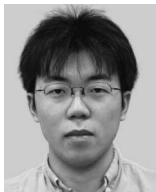
協調型方式におけるチューニング時間 T_{col} は、問合せをサーバに送信する時間 U 、識別子領域を確保するまでの待ち時間 I 、ECAルールの放送にかかる時間 D 、メイン放送帯域から問合せに必要なタプルを受信するためにかかる時間の和となり、以下のように表される。

$$T_{col} = U + I + D + \frac{C_{col}}{g} \quad (30)$$

(平成 15 年 3 月 30 日受付)

(平成 15 年 4 月 3 日採録)

(担当編集委員 清木 康, 遠山 元道)



加下 雅一 (学生会員)

2001 年大阪大学工学部情報システム工学科卒業。同年、同大学院工学研究科博士前期課程入学、現在に至る。放送型通信およびデータベースシステムに興味を持つ。



寺田 努 (正会員)

1997 年大阪大学工学部情報システム工学科卒業。1999 年同大学院工学研究科博士前期課程修了。2000 年同大学院工学研究科博士後期課程中退後、同大学サイバーメディアセンター助手。2002 年より大阪大学大学院情報科学研究科マルチメディア工学専攻助手を併任、現在に至る。アクティブデータベース、モバイルコンピューティング、データ放送の研究に従事。



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情報システム工学専攻助手。2002 年より同大学院情報科学研究科マルチメディア工学専攻助手となり、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。データベースシステム、分散処理に興味を持つ。IEEE、電子情報通信学会、日本データベース学会の各会員。



塚本 昌彦 (正会員)

1987 年京都大学工学部数理工学科卒業。1989 年同大学院工学研究科修士課程修了。同年、シャープ(株)入社。1995 年大阪大学大学院工学研究科情報システム工学専攻講師、1996 年より、同大学大学院工学研究科情報システム工学専攻助教授。2002 年より同大学院情報科学研究科マルチメディア工学専攻助教授となり、現在に至る。工学博士。ウェアラブルコンピューティング、ユビキタスコンピューティング、エンタテインメントコンピューティングに興味を持つ。ACM、IEEE 等 7 学会の会員。



西尾章治郎 (正会員)

1975 年京都大学工学部数理工学科卒業。1980 年同大学院工学研究科博士後期課程修了。工学博士。京都大学工学部助手、大阪大学基礎工学部および情報処理教育センター助教授、大阪大学大学院工学研究科情報システム工学専攻教授を経て、2002 年より同大学院情報科学研究科マルチメディア工学専攻教授となり、現在に至る。2000 年より大阪大学サイバーメディアセンター長を併任。この間、カナダ・ウォータールー大学、ビクトリア大学客員。データベース、マルチメディアシステムの研究に従事。現在、ACM Trans. on Internet Technology, Data & Knowledge Engineering, Data Mining and Knowledge Discovery, The VLDB Journal 等の論文誌編集委員。情報処理学会フェロー含め、ACM、IEEE 等 9 学会の会員。