



放送型データベースシステムにおけるデッドライン を考慮した問合せ処理方式

北島, 信哉
寺田, 努
原, 隆浩
西尾, 章治郎

(Citation)

電子情報通信学会論文誌. D, 情報・システム, J89-D(2):151-162

(Issue Date)

2006-02-01

(Resource Type)

journal article

(Version)

Version of Record

(Rights)

copyright(c)2006 IEICE

(URL)

<https://hdl.handle.net/20.500.14094/90002082>



放送型データベースシステムにおけるデッドラインを考慮した 問合せ処理方式

北島 信哉[†] 寺田 努^{††} 原 隆浩[†] 西尾章治郎[†]

Query Processing Methods Considering the Deadline of Queries for Database Broadcasting Systems

Shinya KITAJIMA[†], Tsutomu TERADA^{††}, Takahiro HARA[†], and Shojiro NISHIO[†]

あらまし 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手法としては、サーバが問合せ処理を行って結果をクライアントに放送する方式、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問合せ処理を行う方式、サーバとクライアントが協調して問合せ処理を行う方式の 3 方式が考えられる。これらの方式は、問合せ発生間隔や問合せ結果サイズ等の環境の変化に応じてその性能に優劣が生じるが、システム環境は常に変化し続けるため静的に最適な方式を選択できない。そこで本論文では、発行された問合せのデッドラインとシステムの状況を考慮して、サーバにおける問合せのキューへの割込み処理や、キュー内の問合せの処理方式の変更を行う問合せ処理手法を提案する。提案方式を用いることで、応答時間をできる限り低減しながら、問合せ処理の成功率を上げることができる。

キーワード 放送型通信, データ放送, データベース, 問合せ処理

1. ま え が き

近年、無線通信技術の発展に伴い、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各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、更に、データアクセスのスループット向上が期待できる。

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], [7], [10]~[12], 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。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。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、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オンデマンド型方式、ク

[†] 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, 吹田市
Department of Multimedia Eng., Graduate School of Information Science and Technology, Osaka University, 1-5 Yamadaoka, Suita-shi, 565-0871 Japan

^{††} 大阪大学サイバーメディアセンターサイバーコミュニティ研究部門,
茨木市
Cybercommunity Division, Cybermedia Center, Osaka University, 5-1 Mihogaoka, Ibaraki-shi, 567-0047 Japan

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問合せ処理を行うクライアント型方式、サーバとクライアントが協調して問合せ処理を行う協調型方式 [8] の 3 方式が考えられる。これらの方式は、問合せ発生間隔や問合せ結果サイズ等の環境の変化に応じてその性能に優劣が生じるが、システム環境は常に変化し続けるため静的に最適な方式を選択することは困難である。

そこで本論文では、発行された問合せのデッドラインとシステム状況を考慮して、サーバにおける問合せのキューへの割込み処理や、キュー内の問合せの処理方式の変更を行い、動的に適切な問合せ処理を実行する手法を提案する。更に、シミュレーション評価により、提案方式が応答時間をできる限り低減しながら、問合せ処理の成功率を上げることを確認する。

以下、2. では放送型データベースシステムについて述べる。3. で提案方式について説明し、4. では提案方式の性能評価を行う。5. で提案方式についての考察と関連研究との比較を行い、6. で本論文のまとめと今後の課題について述べる。

2. 放送型データベースシステム

本研究では、図 1 に示すように、放送型情報システムにおいてサーバがリレーショナルデータベースの内容を送信し、ユーザ（クライアント）が問合せを行う放送型データベースシステムを想定する。放送型データベースシステムは、以下に示す要素から構成される。

[サーバ] 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。また、クライアントからの要求に応じて、問合せ処理を実行する。

[クライアント] 放送を受信するクライアントとしては、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。

[ダウンリンク] サーバからクライアントへの放送帯域は、二つのチャンネルに分割されているものとする。サーバは、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。

[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。クライアントは、このアップリンクを用いて問合せをサーバに送信する。

2.1 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

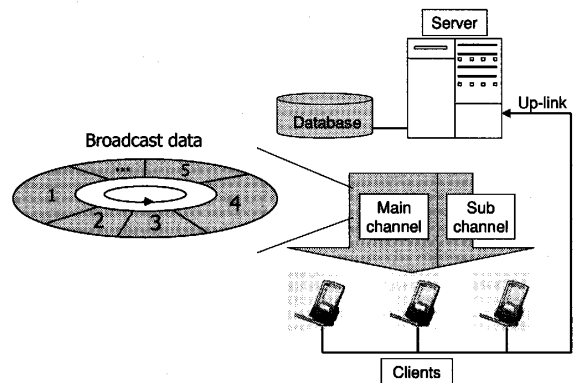


図 1 放送型データベースシステム

Fig. 1 A database broadcasting system.

る。その一例として、ショッピングセンタにおける情報サービスが挙げられる。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千枚、サイズは数百メガバイトとする。ユーザは数千の規模で存在し、各ユーザは常に放送を受信できる環境にあり、放送されている情報を絶えず受信していると想定する。通常、ユーザ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ほしい」といった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。問合せの応答時間は、短いほど好ましい。なお、ユーザは発行した問合せの応答時間を知ることができないため、ユーザが自らで問合せを処理すべきか、サーバが問合せ処理を行った結果を待つべきかの判断はサーバが行うものとする。ユーザは、ショッピングをしながらほしい商品を検索するため、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。ただし、ユーザは各問合せにデッドラインを設定でき、デッドラインの時間内にユーザが問合せの結果が得られない場合は、その問合せは失敗となる。また、放送帯域は 10 Mbit/s 程度とする。放送を受信することのみで問合せを発行しないユーザも存在すると想定し、サーバは、常に同じ放送スケジュールに基づいてデータベースを繰り返し放送し、放送スケジュールのインデックスの配信は行わない。ここで、サーバにおいて放送データの動的なスケジューリングは行わず、すべてのデータ

が一度ずつ放送される放送スケジュールが静的に作成されているとする。

2.2 問合せ処理方式

放送型データベースシステムにおいて、クライアントによる問合せを処理する方式として、以下の3方式がある。

2.2.1 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。

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。クライアントは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、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、応答時間が長くなる可能性がある。

2.2.2 クライアント型方式

クライアントは問合せに関係するすべてのテーブルを自身の記憶領域にいったん蓄え、必要なすべてのデータがそろってから、自ら問合せ処理を行う。

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し、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、クライアントの記憶領域を圧迫する、記憶領域による制約から問合せが処理できない場合がある、クライアントに大きな計算負荷がかかるといった問題点がある。

2.2.3 協調型方式

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とともに、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クライアントに送信する[8]。クライアントは、自分あてに送信された処理ルールをもとに、問合せ結果の作成に必要なタプルの放送開始時刻と放送終了時刻を把握し、その時間に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

```

E: RECEIVE
C: Now.minutes>50
   Now.minutes<70
   New.Q_ID=3
A: STORE(3, "X", New.A)

```

図2 ECA ルールの例

Fig.2 An example of ECA rule.

必要なデータのみを蓄積し、問合せ結果を再現する。

処理ルールの書式には、アクティブデータベースの動作記述言語であるECAルールを用いる。図2に、ECAルールの一例を示す。この例では、クライアントは時刻50から70の間に放送を聞き、テーブルXの識別子3が付けられたタプルの属性Aを蓄積する。

協調型方式では、クライアントは識別子を参照することで、問合せ結果の作成に必要なデータのみを蓄積するため、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できる。また、ECAルールは一般に非常に小さなサイズであるため、オンデマンド型方式に比べてサブ放送帯域の占有時間を短くできる。しかし、各タプルにあらかじめ識別子領域を用意する必要があるため、放送周期が若干長くなる。また、タプルに付加された識別子はクライアントが問合せ結果を作成し終えるまで解放されない。識別子の最大数はあらかじめ決まっているので、問合せが頻繁に起こる場合には識別子が不足し、問合せ成功率が下がってしまう。

3. 問合せ処理方式の選択手法

2.2で述べたオンデマンド型方式、クライアント型方式、協調型方式をそれぞれ単独で用いた場合、問合せ発生間隔や問合せ結果のサイズなどのシステム環境に応じて、その性能に優劣が生じる。そこで、システムの変化に応じて既存の三つの問合せ処理方式の中から最適な方式を選択できれば、システム全体の性能を向上できると考えられる。

そこで本章では、動的に処理方式を選択する手法を提案する。まず応答時間に基づいて単純に方式を選択するLRT方式について述べ、次にその拡張方式について述べる。

3.1 LRT 方式

LRT (Least Response Time) 方式では、クライアントからの問合せがサーバに到着すると、サーバはオンデマンド型方式、クライアント型方式、協調型方式

の各方式を選択した場合の応答時間をそれぞれ計算し、応答時間が最も短い問合せ処理方式を選択する。このとき、クライアント型方式は、問合せ処理に必要なテーブルサイズが、クライアント端末の使用可能なディスク容量よりも大きい場合には使用できない。また、協調型方式は、識別子が不足する場合には使用できない。そのため、応答時間の計算は、使用できる方式についてのみ行う。応答時間が最も短い問合せ処理方式を選択してもデッドラインを越えてしまう場合には、問合せは失敗する。

3.1.1 応答時間の計算

LRT 方式における、オンデマンド型方式、クライアント型方式、協調型方式の各方式における応答時間の計算法を以下に示す。ただし、放送キュー内の k 番目までの送信データサイズの和を $S(k)$ 、サブ放送帯幅を B_s 、クライアントの問合せ発生時刻を T_{gen} と表し、クライアントが問合せをサーバへ送信するのにかかる時間、サーバ及びクライアントが問合せ処理にかかる時間は、データを放送するのにかかる時間と比べて非常に短いため無視する。

[オンデマンド型方式] オンデマンド型方式では、問合せ結果を放送キューの最後尾に追加するため、オンデマンド型方式における応答時間 T_{on} は、サブ放送帯域の放送キューにあるデータをすべて送信し終わるのにかかる時間と、問合せ結果を送信するのにかかる時間の合計となる。サブ放送帯域の放送キューに n 個の送信データが存在するとき、問合せ結果サイズを s_{on} とすると、 T_{on} は次のようになる。ここで s_{on} は、サーバ側で一度問合せ処理を実行することにより求めることができる。

$$T_{on} = \frac{S(n) + s_{on}}{B_s}$$

[クライアント型方式] クライアント型方式における応答時間 T_{cl} は、必要なテーブルがメイン放送帯域ですべて放送されるまでの時間である。サーバでは、クライアントが必要とするテーブルの放送時刻を計出することができる。ここで、クライアントの問合せ発生時刻から最も近い必要テーブル放送開始時刻 $t_{cl}(s)$ は、次の関係を満たす。

$$t_{cl}(s) \geq T_{gen}$$

クライアントは $t_{cl}(s)$ から 1 放送周期以内に全必要テーブルを受信し終わるため、その受信終了時刻を

$t_{cl}(e)$ とすると、時刻 $t_{cl}(e)$ にクライアントは問合せ結果を得る。そのため、 T_{cl} は $t_{cl}(e)$ と T_{gen} の差で表すことができる。

$$T_{cl} = t_{cl}(e) - T_{gen}$$

[協調型方式] 協調型方式では、クライアントはまず、処理ルールを受信した後、処理ルールに従って、識別子の付加されたタプルのみを受信し、問合せ結果を再現する。そのため、協調型方式における応答時間 T_{co} は、サブ放送帯域の放送キューにおいて、処理ルールを挿入する位置よりも前にあるデータをすべて送信し終わるまでにかかる時間と、処理ルールを送信するのにかかる時間、クライアントが処理ルールを受信してから必要タプルを蓄積し終わるまでの時間の合計となる。サーバでは、クライアントが必要とするタプルの放送時刻を計出することができる。放送キューの m 番目に処理ルールを挿入するとき、処理ルールサイズを s_{co} とすると、クライアントの問合せ発生時刻から最も近い必要タプル放送開始時刻 $t_{co}(s)$ は、次の関係を満たす。ここで s_{co} は、サーバ側で一度処理ルールを作成することにより求めることができる。

$$t_{co}(s) \geq T_{gen} + \frac{S(m-1) + s_{co}}{B_s}$$

クライアントは $t_{co}(s)$ から 1 放送周期以内に全必要タプルを受信し終わるため、その受信終了時刻を $t_{co}(e)$ とすると、時刻 $t_{co}(e)$ にクライアントは処理ルールに基づいて問合せ結果を再現する。そのため、 T_{co} は $t_{co}(e)$ と T_{gen} の差で表すことができる。

$$T_{co} = t_{co}(e) - T_{gen}$$

3.1.2 問合せ処理アルゴリズム

LRT 方式における問合せ処理アルゴリズムを以下に示す。

(1) 問合せの発生

クライアントは、SQL で記述された問合せと、端末の記憶容量や問合せのデッドラインなどの付加情報をアップリンクを利用してサーバに送信する。更に、クライアント型方式の問合せ処理に備えて、メイン放送帯域を用いて放送されているデータベースの中から問合せ処理に必要なテーブルを蓄積し始める。

(2) 問合せ処理方式の選択

サーバは、オンデマンド型方式、クライアント型方式、協調型方式のうち、使用できる方式について、そ

れぞれを選択した場合の応答時間を計算し、応答時間が最も短い方式を選択する。

(3) 問合せ処理

サーバは決定した処理方式に基づいて、問合せ処理を実行する。選択された方式がオンデマンド型方式の場合には問合せ処理を行って結果を作成し、クライアント型方式の場合には何も行わない。協調型方式の場合には処理ルールを作成し、放送するタプルに対して識別子を付加する。

(4) クライアントへのデータ送信

サーバは、各方式においてクライアントに配信するデータを、サブ放送帯域の放送キューに追加する。キューに追加されたデータは、先頭から順に放送される。放送されるデータは、オンデマンド型方式では問合せ結果、協調型方式では処理ルールである。クライアント型方式では、サブ放送帯域を用いて放送されるデータはない。

(5) データ受信と問合せ結果の作成

クライアントは、サブ放送帯域から自分あてのデータを受信すると、各方式に基づいて処理を行う。問合せ結果が送信されてきた場合は、クライアントは結果をそのまま利用する。処理ルールが送信されてきた場合は、ルールに基づいて、識別子が付加されたタプルの受信を行い、問合せ結果を再現する。メッセージが送信されない場合は、クライアントは問合せ処理に必要なテーブルの受信が終わり次第、自ら問合せ処理を行う。

LRT方式では、協調型方式における処理ルールをオンデマンド型方式における問合せ結果と同様、キューの最後尾に追加すると、問合せ発生数が多い場合にクライアントが処理ルールを受信するまでの待ち時間が長くなる。協調型方式では、クライアントは処理ルールを受信するまで必要タプルを受信することができないため、結果として協調型方式を選択した場合の応答時間が非常に長くなり、協調型方式の応答時間が短いという利点が失われてしまう。そこで、協調型方式のルールサイズはオンデマンド型方式の問合せ結果サイズに比べて非常に小さいことを考慮して、LRT方式では、サブ放送帯域の放送キューにおいて、協調型方式の処理ルールをキュー内のオンデマンド型方式の問合せ結果よりも前に挿入する。ただし、割り込んだ処理ルールにより、既にキュー内にあるオンデマンド型方式の問合せがデッドラインを越えてしまう場合は、協調型方式を選択できないものとする。

3.1.3 LRT方式の問題点

LRT方式では、問合せがサーバに到着した時点で応答時間が最も短い方式を選択するため、送信データサイズの大きなオンデマンド型方式の選択数が増加すると、キュー内の放送待ちのデータ量が増加する。その結果、デッドラインの制約から、以降の問合せ処理でキューに送信データを追加できない状況が長く続く可能性がある。

また、協調型方式の処理ルールは、キュー内のオンデマンド型方式の送信データの前に割り込むことになるため、以降のオンデマンド型方式の問合せの応答時間がわずかに長くなる。協調型方式の選択数が増加すると、この割込みによる応答時間の増加が無視できなくなる。既にキュー内にあるオンデマンド型方式の問合せがデッドラインを越えることは許さないため、このような状況では、協調型方式を選択できなくなってしまう。

3.2 拡張方式

LRT方式において、オンデマンド型方式、協調型方式、クライアント型方式のうち、使用できる方式のいずれを選択しても応答時間のデッドラインを越えてしまう場合、サーバは問合せ処理方式を選択することができない。そこで、応答時間をできる限り低減し、また、応答時間が増加する場合でも、その増分を最小限に抑え、更に問合せ処理の成功率を上げるために、次の三つの拡張方式を提案する。

拡張方式では、サーバはまずLRT方式に基づいて、オンデマンド型方式、クライアント型方式、協調型方式の中から問合せ処理方式を選択するが、使用できる方式のうち、応答時間が最も短い問合せ処理方式を選択しても応答時間がデッドラインを越えてしまう場合は、特定の処理を実行する。拡張方式における問合せ処理の流れを、図3に示す。

3.2.1 LRT-I方式

LRT方式では、協調型方式を選択した問合せをキューに挿入すると、既にキュー内にあるオンデマンド型方式の問合せの一部の応答時間がデッドラインを越えてしまう場合、協調型方式を選択できなくなってしまう。

LRT-I (LRT-Inserting) 方式では、図4に示すように、協調型方式を選択した際、その処理ルールをキューに挿入することで応答時間がデッドラインを越えてしまうオンデマンド型方式の問合せが存在する場合、その問合せの後ろに処理ルールを挿入する。また、

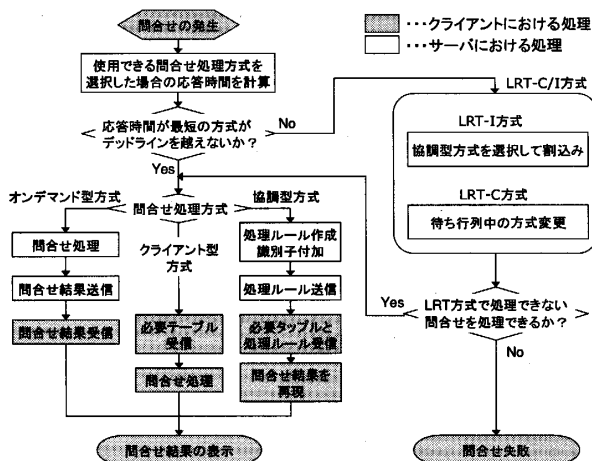


図3 拡張方式における問合せ処理の流れ

Fig. 3 Flow of the query processing in an extended method.

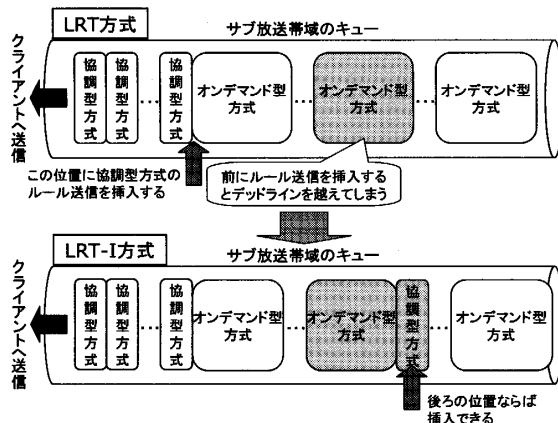


図4 LRT-I方式

Fig. 4 LRT-I method.

そのようなオンデマンド型方式の問合せが複数存在する場合は、該当する問合せのうち、一番後ろにある問合せの後ろに挿入する。このとき、処理ルールを挿入しようとする問合せの応答時間がデッドラインを越えてしまう場合は、問合せは失敗となる。

挿入処理によって問合せの成功率は向上するが、応答時間は長くなる。これは、協調型方式の処理ルールをキューの途中に挿入すると、クライアントが処理ルールを受信するまでの待ち時間が長くなり、協調型方式では処理ルールを受信した後に必要なタプルを受信する必要があることから、協調型方式の応答時間が長くなるためである。

3.2.2 LRT-C方式

LRT方式では、送信データサイズの大きなオンデマンド型方式の選択数が増加すると、キュー内のデー

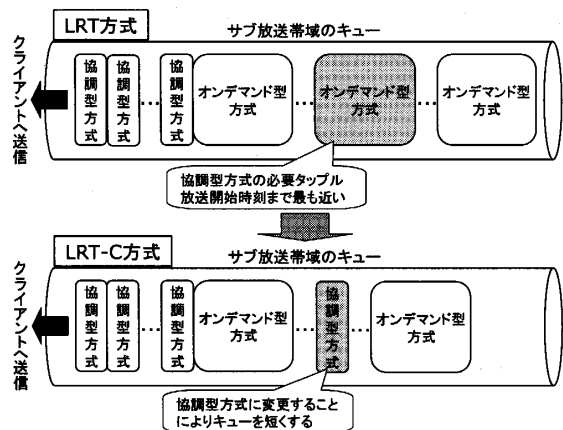


図5 LRT-C方式

Fig. 5 LRT-C method.

タ量が増加し、デッドラインの制約から、以降の問合せ処理でキューに送信データを追加できない状況が続いてしまう可能性がある。

LRT-C (LRT-Change) 方式では、図5に示すように、キュー内のオンデマンド型方式の問合せのうち、協調型方式に変更した場合に、処理ルール受信から必要タプル放送開始時刻までが最も短くなる問合せを、オンデマンド型方式から協調型方式に変更することでキューを短くする。

変更処理は、問合せの成功率を向上させるが、応答時間に関しては長くする要因と短くする要因の両方をもつ。まず、キューに入っている時点でLRT方式により応答時間が最も短い方式が選択されているため、この変更処理によってその問合せ自体の応答時間は長くなる。LRT-C方式では、変更する問合せを上述のように選択することにより、変更処理に伴う応答時間の増加を最小限に抑えている。また、協調型方式における送信データサイズは、オンデマンド型方式における送信データサイズに比べて非常に小さいため、この変更処理によってサブ放送帯域の放送キューが短くなり、放送キューの中で、変更した問合せ以降の問合せの応答時間は短縮される。キュー内の変更した問合せ以降の問合せ数が多い場合や、変更した問合せの問合せ結果サイズが大きい場合にはこの効果が顕著になる。しかし、以降の問合せ処理において、放送キューの前方に協調型方式の処理ルールが挿入されるため、単純に応答時間が短縮されるわけではなく、このような問合せの発生数が多い場合には、結果的に応答時間が長くなる可能性もある。また、変更処理によってキューの後方の問合せにデッドラインまでの余裕が生じるの

で、前方に協調型方式の処理ルールを挿入できるようになり、変更処理を行う前と比べて協調型方式の選択数が多くなる。協調型方式の応答時間は、問合せの発生数が多い場合でも変化せず、オンデマンド型方式に比べて短いため、結果として変更処理以降の問合せの応答時間を低減できる。最終的に、変更処理によって応答時間が長くなるか短くなるかは、状況に依存する。

ただし、この変更処理を行っても問合せが失敗する場合は、変更処理自体を行わない。

3.2.3 LRT-C/I 方式

LRT-I 方式及び LRT-C 方式はそれぞれ独立した処理であるため、これらを組み合わせることにより、更に問合せ成功率を向上できるものと考えられる。

LRT-C/I (LRT-I & LRT-C) 方式は、LRT-I 方式と LRT-C 方式を組み合わせたものである。LRT-C/I 方式では、LRT-I 方式、LRT-C 方式を適用した場合のキュー内のすべての問合せの応答時間の増減を合計し、その値が小さい方式を適用する。また、LRT-I 方式、LRT-C 方式のどちらか一方のみが使用できる場合には、応答時間の増減の計算は行わず、その方式を適用する。ただし、LRT-I 方式では応答時間が長くなるため、LRT-C/I 方式の応答時間は LRT-C 方式に比べて長くなる。

4. 評価

本章では、次に示す二つの評価基準を用いて、LRT, LRT-I, LRT-C, LRT-C/I の各方式の有効性をシミュレーションにより検証する。

- 問合せ成功率

発生した全問合せ数のうち、クライアントが問合せ結果を受け取れたものの割合。

- 平均応答時間

問合せが成功した場合における、クライアントが問合せを発行してから、問合せ結果を受け取るまでの平均時間。ただし、クライアントが問合せをサーバへ送信するのにかかる時間、クライアント及びサーバにおけるデータ処理にかかる時間は、十分に小さいため無視する。

4.1 評価モデル

本評価では、2.1 で示したショッピングセンタ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、データベース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、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,

表 1 評価に用いるパラメータ

Table 1 Parameter configuration.

パラメータ名	値
全タプル数 [個]	4000
1 タプルのサイズ [kByte]	25
1 タプルに付加可能な最大識別子数 [個]	200
メイン放送帯域 [Mbit/s]	10
サブ放送帯域 [Mbit/s]	1
処理ルールのサイズ [kByte]	1
タプル利用率の平均	0.001
タプル利用率の標準偏差	0.00033
記憶領域の不足する端末の割合	0.5

表 2 デッドライン

Table 2 Deadlines.

秒数	60	70	80	90	100	110
割合	5%	5%	10%	20%	20%	10%
秒数	120	130	140	150	160	
割合	10%	5%	5%	5%	5%	

商品名, 画像, ...} をもつものとした。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もち、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もつ。

問合せは SQL によって記述されるものとする。簡単化のため、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する。また、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し、自然結合した結果のタプルには射影演算は行わないものとする。

シミュレーション評価では、各問合せに対し、デッドラインをパラメータとして与え、設定したデッドラインの時間内にクライアントが問合せ結果を受け取れない場合は、問合せは失敗とする。また、クライアント型方式における必要な全テーブルを受信することができない端末の割合をパラメータとして与え、問合せ処理の過程でクライアント側に許容サイズ以上のデータを蓄積しようとした場合は、問合せは失敗とする。

4.2 シミュレーション環境

表 1 に、評価で用いるパラメータとその値を示す。各パラメータは、2.1 で示したショッピングセンタにおける情報サービスを想定して決定した。タプル利用率は、テーブル内の全タプルに対する、問合せ結果に含まれるタプルの割合を表す。評価の際には、各問合せに対し、クライアント型方式における必要テーブルを、クライアントの問合せ発生時刻から 1 放送周期の範囲内でランダムに選択し、選択したテーブ

ルの放送開始時刻と放送終了時刻をパラメータとして与えた。また、協調型方式における必要タプルを、クライアントの問合せ発生時刻から1放送周期の範囲内で問合せ結果サイズに応じてランダムに選択し、選択したタプルの放送開始時刻と放送終了時刻をパラメータとして与えた。更に、問合せをサーバに送信する際、クライアントは60秒から160秒の間でデッドラインを選択できるものとし、その選択割合を表2に示すように設定した。

サーバには、指数分布に従った間隔で問合せが到着するものとし、1秒間当りの問合せ発生数をパラメータとして与えた。

4.3 シミュレーション結果

4.3.1 問合せ成功率の比較

1秒間当りの問合せ発生数を変化させたときの、LRT, LRT-I, LRT-C, LRT-C/Iの各方式と、オンデマンド型方式、クライアント型方式、協調型方式をそれぞれ単独で用いた場合における問合せ成功率を図6に示す。

オンデマンド型方式では、1秒間当りの問合せ発生数が非常に少ない場合の問合せ成功率は最も高いが、問合せ発生数が増加するとともに、問合せ成功率は大きく低下している。これは、オンデマンド型方式では狭帯域のサブ放送帯域を用いて、データサイズの大きな問合せ結果をクライアントに送信するため、問合せ発生数が増加すると、サブ放送帯域が枯渇してしまうからである。

クライアント型方式では、1秒間当りの問合せ発生数が変化しても、問合せ成功率は変化しない。これは、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そろってからクライアント上で問合せ処理を行うため、問合せ発生数が変化しても、システム全体に影響を与えないためである。しかし、クライアントの記憶領域による制約から、問合せ成功率は低くなっている。

協調型方式では、1秒間当りの問合せ発生数が少ない場合の問合せ成功率は高いが、1秒間当りの問合せ発生数が3.5を超えるあたりから、問合せ成功率は大きく低下している。これは、問合せ発生数が増加すると、識別子が枯渇してしまうためである。協調型方式における最長応答時間は最短のデッドラインよりも長い場合、問合せ成功率は100%にならない。

LRT方式では、1秒間当りの問合せ発生数が3から4.5の間では協調型方式よりも問合せ成功率が低い、

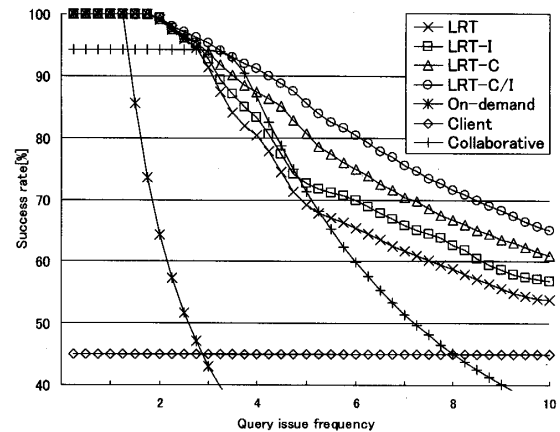


図6 1秒間当りの問合せ発生数と問合せ成功率

Fig. 6 Query issue frequency vs. success rate.

その他の領域では問合せ成功率が向上している。LRT方式ではシステム状況に応じてオンデマンド型方式、クライアント型方式、協調型方式の中から応答時間が最短となる問合せ処理方式を選択することで、3方式を単独で用いる場合よりも問合せ成功率を向上している。しかし、LRT方式では協調型方式の処理ルールの送信を、サブ放送帯域の放送キューの前方に挿入するが、既にキュー内にあるオンデマンド型方式の問合せがデッドラインを越えることは許さないため、協調型方式単独の場合よりも問合せ成功率が下がる場合がある。

LRT-I方式では、LRT方式よりも問合せ成功率が高くなっている。これは、LRT方式を用いた場合に失敗する問合せを、LRT-Iでは協調型方式の挿入位置を変更することで処理できる場合があるからである。同様に、LRT-C方式も、LRT方式より問合せ成功率が向上している。LRT-Cではキュー内のオンデマンド型方式の問合せ処理を協調型方式に変更するため、協調型方式の選択数が増加する。協調型方式における送信データサイズは、オンデマンド型方式における送信データサイズに比べ非常に小さいため、サブ放送帯域の枯渇を緩和し、結果として問合せ成功率が向上する。また、LRT-Iはキューに問合せを追加するのみの処理である一方、LRT-Cはキュー内のデータ量を減少させる効果ももつことから、処理できる問合せ数が増加し、LRT-Iよりも問合せ成功率の上昇幅が大きくなる。

LRT-C/I方式は、両方式の利点を組み合わせて用いるため、最も問合せ成功率が高くなっている。ただし、識別子数には限りがあるため、協調型方式の選択

割合の増加は LRT-C に比べてわずかであり、飛躍的に問合せ成功率が上がることはないと考えられる。

4.3.2 平均応答時間の比較

1 秒間当りの問合せ発生数を変化させたときの、LRT, LRT-I, LRT-C, LRT-C/I の各方式と、オンデマンド型方式、クライアント型方式、協調型方式をそれぞれ単独で用いた場合における平均応答時間を図 7 に示す。

オンデマンド型方式では、1 秒間当りの問合せ発生数が非常に少ない場合には平均応答時間は最も短いが、問合せ発生数が増加するとともに、平均応答時間が非常に長くなっている。これは、問合せ発生数が増加すると、サーバのキュー内のデータが非常に多くなるためである。

クライアント型方式では、1 秒間当りの問合せ発生数が変化しても、平均応答時間は変化しない。これは、問合せ発生数が変化しても、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蓄積するまでの時間は変化しないからである。

協調型方式においても、1 秒間当りの問合せ発生数の変化は、平均応答時間に影響しない。協調型方式では、サブ放送帯域を用いて処理ルールを送信するが、処理ルールのサイズは非常に小さいため、オンデマンド型方式のようにキュー内のデータ量が増加することはない。また、協調型方式では必要なタプルのみを受信すればよいので、クライアント型方式と比べ、平均応答時間は短くなる。

LRT 方式では、1 秒間当りの問合せ発生数が非常に少ない場合に、オンデマンド型方式と同様、平均応答時間が最も短くなっている。また、問合せ発生数の増加とともに平均応答時間は長くなるが、1 秒間当りの問合せ発生数が非常に多い場合でも、クライアント型方式と大きくは変わらない。LRT 方式ではシステムの状態に応じてオンデマンド型方式、クライアント型方式、協調型方式の中から応答時間が最短となる問合せ処理方式を選択するので、問合せ発生数が非常に小さい場合には、LRT 方式においてもオンデマンド型方式のみが選択される。その結果、応答時間はオンデマンド型方式と同様に、最短となる。問合せ発生数が増加すると、協調型方式の選択数が増加するが、オンデマンド型方式も選択される。問合せ発生数が多い場合にはオンデマンド型方式の応答時間は非常に長くなるため、LRT 方式における応答時間は協調型方式よりも長くなる。更に問合せ発生数が増加すると、クラ

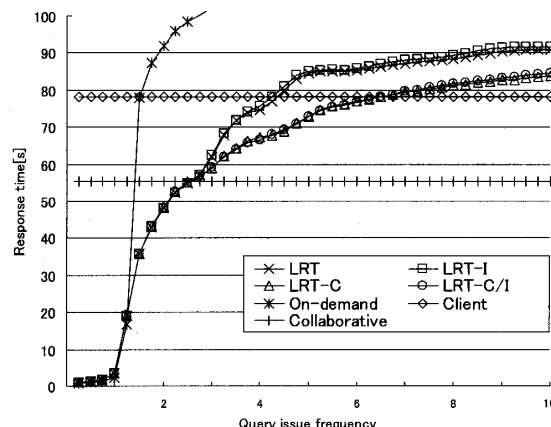


図 7 1 秒間当りの問合せ発生数と応答時間

Fig. 7 Query issue frequency vs. response time.

イアント型方式の選択数が増加するが、同様にオンデマンド型方式も選択されるため、LRT 方式における応答時間はクライアント型方式よりも長くなる。

LRT-I 方式では、LRT 方式よりも平均応答時間がやや長くなっている。これは、LRT 方式においてキューの前方に挿入する協調型方式の処理ルールを、LRT-I ではキューの途中にも挿入することにより、協調型方式の応答時間が長くなるためである。

LRT-C 方式では、LRT 方式よりも平均応答時間が短くなっている。LRT-C では、キューに挿入される際に、LRT で応答時間が最も短い問合せ処理方式が選択されているため、処理方式を変更することによって、その問合せの応答時間は変更前に比べて長くなる場合がある。しかし、協調型方式における送信データサイズは、オンデマンド型方式における送信データサイズに比べて非常に小さいため、この変更処理によってサブ放送帯域の放送キューが短くなり、放送キューの中で、変更した問合せ以降の問合せの応答時間が短縮される。更に、この変更処理によってキューの後方の問合せにデッドラインまで余裕が生じるので、前方に協調型方式の処理ルールを挿入することができるようになり、以降は応答時間の短い協調型方式の選択数が多くなる。本実験では、応答時間が短縮される要因の方が大きかったため、結果的に平均応答時間が短くなったものと考えられる。

LRT-C/I 方式では、LRT-I 方式と LRT-C 方式を選択的に使用する。LRT-C 方式では平均応答時間が短くなるが、LRT-I 方式では平均応答時間は長くなるため、結果として LRT-C/I の平均応答時間は LRT-C に比べてやや長くなる。

5. 考 察

5.1 問合せ処理方式の選択について

シミュレーション結果では、1秒間当りの問合せ発生数が少ない場合、LRT方式でも十分な性能を示している。また、1秒間当りの問合せ発生数が多い場合には、LRT-C/I方式の問合せ成功率が最も高いが、平均応答時間はLRT-C方式の方が短い。システム環境によって問合せ発生数は異なり、また、問合せ成功率の向上と平均応答時間の低減を両立することは難しいことから、どの提案方式が最も性能がよいかは、一概に判断できない。そのため、問合せ発生数が少ない場合にはLRT方式を使用し、問合せ発生数が多い場合には、問合せ成功率を重要視するユーザが多い状況ではLRT-C/I方式を使用し、平均応答時間を重要視するユーザが多い状況ではLRT-C方式を使用するということに、状況に応じて提案方式を使い分ける必要がある。

状況に応じた方式の動的な切替に関しては今後の課題であるが、クライアントが問合せをサーバに送信する際に、問合せ成功率と平均応答時間のどちらを優先するかを選択して問合せとともにサーバに送信し、サーバはその選択割合から使用する方式を決定するという方法が考えられる。

5.2 バースト的な問合せ発生について

問合せがバースト的に発生する状況下では、オンデマンド型方式の送信データがキュー内に存在すると、サブ放送帯域の枯渇が顕著に現れる。提案方式ではこの現象が多少改善されるものの、限界があるため、このような状況下ではオンデマンド型方式は選択すべきではない。

バースト的な問合せに対する選択アルゴリズムの性能向上については今後の検討課題であるが、この問題を解決する方法として、サーバが1秒間当りの問合せ発生数の平均を統計的に求め、あらかじめ設定したしきい値を超えた場合は、オンデマンド型方式を選択しないという方法が考えられる。

5.3 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。この研

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一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],[6]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]では放送チャネルを分割し、プッシュ型、プル型通信チャネルの最適な帯域分割について論じている。また、文献[6]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、プル型の放送チャ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、問合せ結果や処理ルールを放送することで、より柔軟かつ効果的にプッシュ型チャネルを利用している点で、これらの研究とは異なる。

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) [9]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。

6. む す び

本論文では、放送型データベースシステムにおいて、問合せのデッドラインを考慮して、動的に問合せ処理方式を選択する手法を提案した。提案方式では、放送型データベースシステムに対する三つの処理方式の中から応答時間が最も短いものを選択する。更に提案方式の拡張方式では、サーバの放送キューに注目し、キューへの割込み処理やキュー内の問合せの処理方式の変更を行うことで、問合せ処理の効率を高めている。また、提案方式の有効性を検証するために、問合せ成功率と平均応答時間についてシミュレーション評価を

行った。シミュレーション評価の結果から、提案方式は従来方式と比べて応答時間をできる限り低減しながら、問合せ処理の成功率を上げることができることを確認した。

今後は、単位時間当りの問合せ発生数に応じて、問合せ処理方式を動的に選択する手法について検討する必要がある。更に、時間とともに問合せの発生間隔が変化する環境において、より適切な問合せ処理方式を選択する手法についても検討する必要がある。

謝辞 本研究の一部は、文部科学省 21 世紀 COE プログラム「ネットワーク共生環境を築く情報技術の創出」、及び文部科学省特定領域研究 (16016260)、基盤研究 (A)(17200006)、基盤研究 (B)(2)(15300033) の研究助成によるものである。ここに記して謝意を表す。

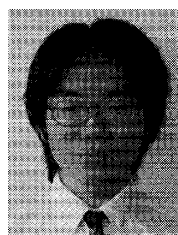
文 献

- [1] S. Acharya, M. Franklin, and S. Zdonik, "Broadcast disks: Data management for asymmetric communication environments," Proc. ACM SIGMOD'95, pp.199-210, May 1995.
- [2] S. Acharya, M. Franklin, and S. Zdonik, "Disseminating updates on broadcast disks," Proc. VLDB'96, pp.354-365, Sept. 1996.
- [3] S. Acharya, M. Franklin, and S. Zdonik, "Balancing push and pull for data broadcast," Proc. ACM SIGMOD'97, pp.183-194, May 1997.
- [4] D. Aksoy and M. Franklin, "Scheduling for large-scale on-demand data broadcasting," Proc. IEEE INFOCOM, pp.651-659, March 1998.
- [5] D. Aksoy, M. Franklin, and S. Zdonik, "Data staging for on-demand broadcast," Proc. VLDB'01, pp.571-580, Sept. 2001.
- [6] 箱根 聡, 田辺雅則, 石川裕治, 井上 潮, "放送型通信とオンデマンド型通信を統合した情報提供システム," 情処学論, vol.40, no.10, pp.3772-3781, Oct. 1999.
- [7] Q. Hu, D. Lee, and W. Lee, "Performance evaluation of a wireless hierarchical data dissemination system," Proc. Mobicom'99, pp.163-173, Aug. 1999.
- [8] 加下雅一, 寺田 努, 原 隆浩, 塚本昌彦, 西尾章治郎, "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," 情処学論データベース, vol.44, no.SIG8 (TOD 18), pp.92-104, June 2003.
- [9] 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の設計と実装," 信学論 (D-I), vol.J83-D-I, no.12, pp.1272-1283, Dec. 2000.
- [10] E. Yajima, T. Hara, M. Tsukamoto, and S. Nishio, "Interval optimization of correlated data items in data broadcasting," Proc. International Conference on Advances in Information Systems (ADVIS'00), pp.127-136, Oct. 2000.
- [11] E. Yajima, T. Hara, M. Tsukamoto, and S. Nishio,

"Scheduling strategies of correlated data in push-based systems," Information Systems and Operational Research (INFOR), vol.39, no.2, pp.152-173, May 2001.

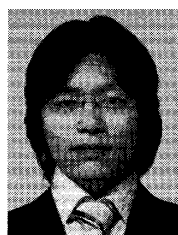
- [12] E. Yajima, T. Hara, M. Tsukamoto, and S. Nishio, "Scheduling and caching strategies for broadcasting correlated data," Proc. ACM Symposium on Applied Computing (ACM SAC'01), pp.504-510, March 2001.

(平成 17 年 5 月 11 日受付, 9 月 1 日再受付)



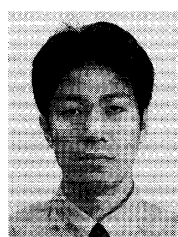
北島 信哉

平 17 阪大・工・電子情報エネルギー卒。現在、同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在学中。放送型通信及びデータベースシステムに興味をもつ。日本データベース学会学生会員。



寺田 努 (正員)

平 9 阪大・工・情報システム卒。平 11 同大学院工学研究科修士課程了。平 12 同大学院退学。同年より大阪大学サイバーメディアセンター助手。平 17 より同講師、現在に至る。平 14 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、平 17 より同講師を併任。工博。アクティブデータベース、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の研究に従事。平 15 年度本会論文賞受賞。IEEE, 情報処理学会, 日本データベース学会各会員。



原 隆浩 (正員)

平 7 阪大・工・情報システム卒。平 9 同大学院工学研究科博士前期課程了。同年同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、同大学院情報科学研究科マルチメディア工学専攻助手を経て、平 16 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。工博。データベースシステム、モバイルネットワーク、分散処理の研究に従事。平 8 情報処理学会山下記念研究賞、平 12 電気通信普及財団テレコムシステム技術賞各受賞。情報処理学会, ACM, IEEE 各会員。



西尾章治郎 (正員：フェロー)

昭 50 京大・工・数理卒。昭 55 同大大学院工学研究科博士後期課程了。工博。京大工学部助手、阪大基礎工学部及び情報処理教育センター助教授、阪大大学院工学研究科情報システム工学専攻教授を経て、平 14 より同大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。平 12 より阪大サイバーメディアセンター長、平 15 より阪大大学院情報科学研究科長を併任。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、Data & Knowledge Engineering 等の論文誌編集委員。本会平 13 年度業績賞、平 15 年度論文賞各受賞。情報処理学会フェローを含め、ACM、IEEE など 8 学会各会員。