[B3–1] 配信型情報源統合環境における統合要求記述の正当性の検証

渡辺 陽介 † 石川 佳治 †† 北川 博之 ††

† 筑波大学 システム情報工学研究科

†† 筑波大学 電子・情報工学系

{watanabe, ishikawa, kitagawa}@kde.is.tsukuba.ac.jp

概要

近年、ネットワーク上で多様な情報源が利用可能であり、異種情報源統合はデータ工学の重要な研究分野の一つとなっている。また、新しい情報源として、メールマガジンやデータ放送のような配信型情報源が注目されている。我々の研究グループは、既存の配信型情報源を統合して新たな情報配信サービスを定義することのできる、配信型情報源統合環境の構築を行なってきた。この統合環境では、ユーザが与えた代数式ベースの配信要求記述から、情報の到着などのイベントに応じた処理を実現するために必要な ECA ルールを自動生成している。本稿ではユーザが与えた配信要求記述を検証するための手法を中心に述べる。実現可能な配信要求記述が持つべき性質として整合性と充足可能性を定義し、それらの性質を検証するための手法について検討する。

1 はじめに

ネットワークの発達にともない、各種情報源へのアクセスが容易になってきた。そのため、異種情報源の統合利用への要求が増大しており、データ工学においては情報統合が重要な研究課題の一つとなっている [1, 2, 3, 4, 5]。また、情報源の形態も多様化し、その中の１つとして配信型情報源が注目されている。配信型情報源は、情報配信サーバからクライアントに対して情報を能動的に配信することを特徴とする情報源であり、これによってユーザは、情報がどこにあり、いつ更新されたかを気にすることなく新鮮な情報を素早く入手することが可能となる。配信型情報源の代表的な例として、プッシュ型配信サービス [6, 7, 8]、データ放送 [9, 10]、メールマガジンなどがあげられる。

配信型情報源の統合においては、複数の配信型情報源や他の情報源の情報を統合して新たな配信サービスを提供したり、ユーザが希望する配信タイミングで情報を配信するといったことが望まれる。この際、配信型情報源では情報の配信は随時行われるため、配信型情報源の統合利用を実現するには、情報の到着や時間の経過などのイベントに起因して能動的に情報の再構成や配信処理を行う必要がある。我々の研究グループは、ECA ルール [11] を記述することで、イベントに起因する能動的な情報の再構成や配信処理を行うことができる配信型情報源統合環境を構築してきた [12]。しかし、配信型情報源の統合を実現するために必要な ECA ルールは通常複数であり、ユーザがこれら複数のルールをルール間の関係まで意識しながら記述することは容易ではない。この問題に対する１つのアプローチとして、ユーザが与えた配信要求記述から、必要な ECA ルールを自動的に生成することが考えられる。我々は配信型情報源をリレーションとしてモデル化し、リレーショナル代数の枠組みをベースとした配信要求の記述方法を定義した。また、そのような代数記述から自動的に ECA ルールを生成する方法を提案した [13]。

一般に、放送局のような配信型情報源は配信スケジュールに基づいて情報を送信している。したがって、配信要求記述だけを調べて ECA ルールを生成するよりも、情報源が持つ固有の性質を考慮した方が、配信要求記述の正当性を検証したりルール生成上の無駄を省く上で有利であると考えられる。[14] では [13] の枠組みを拡張し、配信型情報源から情報が送られてくるタイミングに関する性質を考慮した配信要求記述の処理方式を提案した。さらに [14] では、ユーザの要求が妥当か否かを検証するための基準として、整合性と充足可能性の概念を定義した。整合性は、ユーザに配信する統合結果が配信時刻までに到着した情報のみを使って生成できることを保証する。充足可能性は、ユーザが配信を要求する情報が実際に存在することを保証する。本稿では、整合性、充足可能性を検証するためのより具体的な手法を中心に述べる。

まず、2 節で情報源を統合して新たな情報配信サービスを定義する例を示す。3 節では本稿の前提となる配信型情報源統合環境の概要について説明する。4 節で配信型情報源から情報が配信されるタイミングに関する性質の記述法について述べる。5 節で配信要求記述の整合性と充足可能性の概念について説明し、6 節でそれらの検証法について述べる。7 節では整合性や充足可能性を考慮したルール生成の概要を述べる。8 節で関連研究について述べ、9 節でまとめと今後の課題について述べる。

2 統合例

本節では、２種類の配信型情報源とリレーショナルデータベースから新たな情報配信サービスを定義する統合例を示す。まず以下のような情報源を仮定する (図 1)。

- **サッカー情報配信サービス (Soccer):**
  この情報源はその週の週末に行われるサッカーの試合に関する情報 (**配信単位**) を配信している。送られてくる配信単位は、試合の日付 (date)、試合会場 (place)、チーム名 (teams)、より詳細な内容 (moreInfo) などを含んでいる。この情報源からの情報は毎週水曜日に届くものとする。
- **天気予報配信サービス (Weather):**
  この情報源は天気予報の配信単位を地域毎にわけて配信している。情報が届くのは毎日午前６時であるとする[1]。送られてくる天気予報は、今後７日分の週間予報と４週間分の長期予報に分けられている。週間予報にはそれぞれ配信単位ごとに、日付 (date)、地域 (location)、天気 (weather)、降水確率 (precipitation) の情報が含まれている。長期予報には、１週間ごとにその週の天気の傾向が格納されている。

[1]実際には全ての配信単位が配信されるまで一定の時間を必要とすると考えられるが、ここでは議論の簡単化のためこのように仮定する。

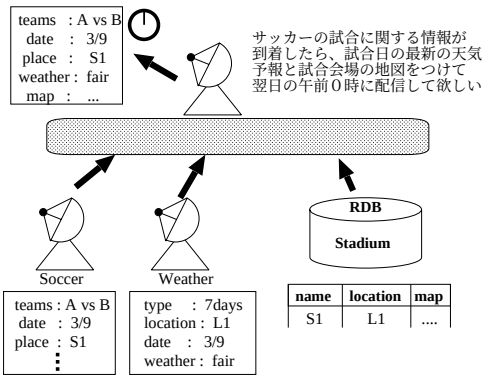

図 1: 統合例

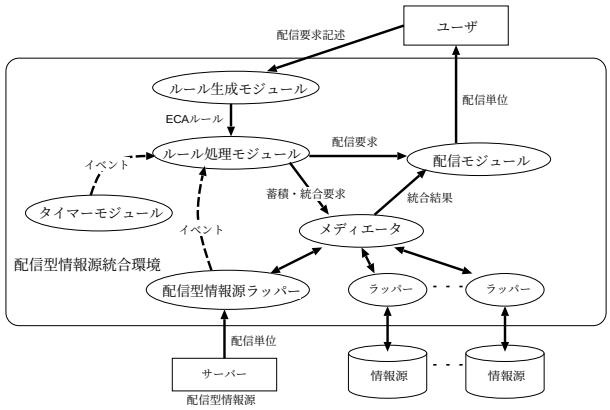

図 2: 統合システムアーキテクチャ

- **競技場情報リレーション (Stadium):**
  サッカーの試合会場に関する情報源。この情報源は配信型情報源ではなく、通常のリレーショナルデータベースであるとする。属性として施設名 (name)、所在地 (location)、電話番号 (phone)、地図 (map) などの情報を格納している。

上記の情報源に対する統合要求の例として、「サッカーの試合に関する情報が到着したら、試合日の最新の天気予報と試合会場の地図をつけて、翌日の午前 0 時に配信して欲しい」というものを考える。これは、サッカー情報配信サービス、天気予報配信サービスという 2 つの配信型情報源と競技場情報リレーションという非配信型情報源を統合し、統合結果を周期的に送信するという、新たな配信型情報サービスを定義することに相当する。

3 統合システム概要

本節では、本研究の前提となる配信型情報源統合環境 [12, 13] について説明する。

3.1 アーキテクチャ

本配信型情報源統合環境は、統合アーキテクチャとしてメディエータ／ラッパー方式を採用している (図 2)。まず情報源ごとに対応する**ラッパー**が存在し、情報源に固有な変換処理を請け負っている。ラッパーの上位には**メディエータ**があり、情報の統合処理を行う。本システムは統合における共通データモデルとしてリレーショナルモデルを用いている。各情報源のデータ構造をリレーションに変換する作業はラッパーが行う。

配信型情報源には、**配信型情報源ラッパー**を用いる。配信型情報源ラッパーは、通常のラッパーとしての機能に加え、情報配信サーバから配信単位が送られてくると、情報が到着したことを通知する **arrival イベント**を発生する機能を持つ。イベントを発生させるモジュールとして、他にタイマーモジュールがあり、指定した時刻が来たことを通知する **alarm イベント**を発生する。

発生したイベントは ECA ルールを管理する**ルール処理モジュール**に通知される。ECA ルールは、イベントに応じた処理を行うためのものである。イベントが発生すると、ルール処理モジュールはイベントに対応する ECA ルールを評価・処理し、ルールに従ってメディエータへの統合処理要求や、**配信モジュール**への統合結果の配信要求を行う。

本統合環境では、ユーザが ECA ルールを直接記述する必要はなく、代わりに配信要求記述を与えることで新たな配信サービスを定義できる。**ルール生成モジュール**は、ユーザから渡された配信要求記述から統合・配信処理に必要な ECA ルールを自動生成する。配信要求記述の詳細は 3.3 節で説明する。ECA ルール生成の概要は 7 節で述べるが、詳細については [13, 14] を参照されたい。

2 節での例を用いて各モジュールの動作を説明する。

1. ユーザが配信要求記述をルール生成モジュールに与えると、必要な ECA ルールが生成されてルール処理モジュールに渡される。
2. 配信型情報源ラッパーが天気予報配信サービスから配信単位を受け取ると、ラッパーはそれをリレーションのタプルに変換し、ルール処理モジュールに情報の到着を表すイベント（arrival イベント）を通知する。
3. ルール処理モジュールは通知された arrival イベントに対応する ECA ルールを発火し、到着した配信単位を一時リレーションに蓄積するようメディエータに要求する。
4. サッカー情報配信サービスから配信単位が到着すると、配信型情報源ラッパーは同様に arrival イベントを発生する。
5. ルール処理モジュールは別の ECA ルールを発火し、サッカー情報配信サービスから到着した情報を別の一時リレーションに蓄積するようメディエータに要求する。さらに、翌日の午前 0 時にタイマーをセットする。
6. 午前 0 時になると、タイマーモジュールは alarm イベントを発火する。ルール処理モジュールはそのイベントを受け取ると、サッカー情報と同じ日に届いた天気予報、さらに施設情報を統合して新たな配信単位を生成するようメディエータに要求する。
7. 最後に、ルール処理モジュールは配信モジュールに統合結果の新たな配信単位をユーザに配信するように要求する。

3.2 ECA ルール

ECA ルールはアクティブデータベース [11] で使われているものと同様で、イベントに応じた処理を記述するのに用いる。各ルールはイベント節 (on 節)、コンディション節 (if 節)、アクション節 (do 節) で構成される。

イベント節には、ルールが発火するためのきっかけとなるイベントを記述する。配信単位の到着を表す arrival イベントが配信型情報源ラッパーによって生成され、指定された時刻になったことを知らせる alarm イベントがタイマーモジュールによって生成される。

コンディション節には、アクション節に記述された処理が実行されるために満たされるべき条件を記述する。本研究では、コンディション節に記述される条件としてリレーショナル代数演算の選択条件のみを許す。

アクション節には、情報の蓄積、統合、配信の処理を記述する。これらのアクションは基本的にリレーショナル代数式を用いて記述する。

3.1 節で述べたように、配信型情報源を統合して新たな情報配信サービスとして提供するためには、配信単位の蓄積、統合、配信などの処理が必要となる。本研究では、これら一連の処理を以下の 3 つのフェーズに分け、それぞれのフェーズに対応する処理を行なうための ECA ルールを生成することによって配信要求を実現している。

1. **配信単位の蓄積**
   このフェーズは arrival イベントによって発火する。到着した配信単位がユーザの要求を満たすのに必要なら、メディエータ内の一時リレーションに蓄積する。この処理を行うルールを**蓄積ルール** (storage rule) と呼ぶ。
2. **新たな配信情報の生成と配信**
   このフェーズは alarm イベントによって発火し、情報を統合して要求された新たな配信単位を生成・配信する。この処理を行うルールを**生成ルール** (generation rule) と呼ぶ。
3. **不要なデータの廃棄**
   一時リレーションから将来使用されないことが明らかな情報を定期的に削除する。この処理を行うルールを**廃棄ルール** (garbage disposal rule) と呼ぶ。このルールは alarm イベントによって発火する。

例えば、3.1 節のステップ 3 に対応する蓄積ルールは以下のように書ける。

Rule $Storage_{Weather}$
on: $arrival_{N_{Weather}}$
if: $N_{Weather}.type =$ '7 days'
do: $Temp_{Weather} += N_{Weather}$

ここで、$N_{Weather}$ は天気予報配信サービスから配信された最新の配信単位を表し、$Temp_{Weather}$+=$N_{Weather}$ は、一時リレーション $Temp_{Weather}$ に到着した配信単位を追加することを表す。

3.3 配信要求記述

本統合環境は各情報源をリレーションとしてモデル化しており、配信型情報源も同様にリレーションとして表される。情報配信サーバから到着した配信単位は、そのリレーションに属する 1 タプルとしてモデル化される。我々は統合対象となるこのようなリレーションを **I-sequence リレーション**と呼んでいる。I-sequence リレーションは、その情報源が元々持っている属性の他に、タプルが到着した時刻を表すための **ITS 属性**を持つ。統合結果も同様にリレーションとして表され、統合結果となる各タプルは配信予定時刻が来るとユーザに配信される。この統合結果のリレーションを **O-sequence リレーション**と呼ぶ。O-sequence リレーションは、各タプルの配信予定時刻を表す **OTS 属性**を持つ。ITS・OTS 属性ともに時刻ドメインの値である。

上記のようなモデル化をしたことで、ユーザが要求する統合結果の O-sequence リレーションを、I-sequence リレーションや他の情報源のリレーションからどのようにして得るかをリレーショナル代数で記述することにより、新たな情報配信サービスを定義することが可能となる。

具体的には、配信要求記述は以下のような式で定義する。

$$O_{new} = \Omega_{f(I_k.ITS)}(E(I_1,\ldots,I_n)),$$

O_{new} は O-sequence リレーション、E は I-sequence リレーション $I_1,\ldots,I_n$ から新たなリレーションを生成するためのリレーショナル代数式である[2]。また、$\Omega_{f(I_k.ITS)}$ は新たに導入した演算子で、E を評価して得られるリレーションに OTS 属性を付加してその値を $f(I_k.ITS)$ に設定し、ITS 属性を全て除去する操作を表す。関数 f については以下に述べる。

本研究ではリレーショナル代数式 E について、以下のような仮定を行う[3]。

- E はリレーショナル代数の結合演算、直積演算、選択演算、射影演算からなる代数式であるとする。
- Ω が ITS 属性を利用するので、E の中では ITS 属性に対する射影演算は行われないとする。
- E 中のタイムスタンプに関する選択・結合条件では、後述するタイムスタンプ関数と比較演算 $(=,<,>,\leq,\geq)$ のみを使用可能とし、これらの条件は AND で結合されているとする。

関数 f は**タイムスタンプ関数**で、リレーション I_k の ITS 属性の値から新たなタイムスタンプを生成する。I_k を**マスタ I-sequence リレーション** (または**マスタ情報源**) と呼び、E 中で参照されていなければならない。タイムスタンプ関数として、以下の 5 つの関数が使用可能であるとする。

1. **immediate**(**t**): 与えられた t そのものを返す。
2. **next**$_p$(**t**): t に最も近い未来の条件 p を満たす時刻を返す。
3. **previous**$_p$(**t**): t に最も近い過去の条件 p を満たす時刻を返す。
4. **after**$_{\delta t}$(**t**): t から時間 δt だけ経過した時刻
5. **before**$_{\delta t}$(**t**): t から時間 δt だけ遡った時刻

時刻の条件 p と δt はそれぞれ以下の形式で記述する。

p → dayofweek:hour:minute:second

δt → day:hour:minute:second

[2] E の中では、通常のデータベース中のリレーションなど I-sequence リレーション以外の他のリレーションも参照可能であるが、以下の議論ではこれらは本質的でないので、ここでは明記していない。

[3] 和演算や条件式中の OR を含めても本質的な問題はないが、ここでは議論を簡単にするためこのように仮定する。

day、hour、minute、second の各フィールドには非負整数かワイルドカード (*) を記述することができる (δt にはワイルドカードは指定できない)。フィールド dayofweek には、文字列 {*Sun, Mon, Tue, Wed, Thu, Fri, Sat* } のうちの一つか、ワイルドカードを指定できる。

タイムスタンプ関数の利用例として

$$next_{*:0:0:0}(Soccer.ITS)$$

は Soccer の配信単位の到着した時刻の次の午前 0 時に対応するタイムスタンプを返す。

この枠組に従うと、2 節で説明した統合例に対する配信要求は以下のような代数式で記述することができる。

$$\begin{aligned}
O_{new} = \Omega_{next_{*:0:0:0}(Soccer.ITS)}(&\\
&Soccer\\
&\bowtie_{Soccer.date=Weather.date}\\
&\quad \wedge previous_{*:0:0:0}(next_{*:0:0:0}(Soccer.ITS)) \leq Weather.ITS\\
&\quad \wedge Weather.ITS < next_{*:0:0:0}(Soccer.ITS)\\
&\sigma_{Weather.type=\text{`7 days'}}(Weather)\\
&\bowtie_{Weather.location=Stadium.location}\\
&\quad \wedge Soccer.place=Stadium.name\\
&Stadium\\
),&
\end{aligned}$$

ただし、*Soccer* と *Weather* は I-sequence リレーションであり、$\sigma, \bowtie$ は選択演算と結合演算である。以下の結合条件は、サッカー情報と同じ日に届いた週間予報を統合することを表している。

$$\begin{aligned}
&previous_{*:0:0:0}(next_{*:0:0:0}(Soccer.ITS)) \leq Weather.ITS\\
&\wedge Weather.ITS < next_{*:0:0:0}(Soccer.ITS)
\end{aligned}$$

4 配信型情報源の性質記述

本節では、配信型情報源から配信単位が送られるタイミングに関する性質を記述する方法について説明する。ここでは制約記述言語 [15] を元にした表記法を用いる。配信型情報源の性質記述は以下のような式で表される。

$$I_i \equiv \sigma_{PROP_i}(I_i)$$

σ はリレーショナル代数における選択演算である。この式は、左辺と右辺が同値であることを保証し、式中で情報源 I_i が参照されていた場合に、それを右辺で置き換えてもよいことを宣言するものである。*PROP* の部分に *ITS* 属性に関する条件を記述することにより、配信型情報源の性質記述を行うことができる。

性質記述の例を示す。2 節の例では、2 つの配信型情報源が登場した。まず、サッカー情報配信サービスでは情報が毎週水曜日に届くと仮定した。よって、以下のように性質を記述することができる。

$$\begin{aligned}
&Soccer \equiv\\
&\quad \sigma_{previous_{Wed:0:0:0}(Soccer.ITS) \leq Soccer.ITS\\
&\quad \wedge Soccer.ITS < after_{1:0:0:0}(previous_{Wed:0:0:0}(Soccer.ITS))}(Soccer)
\end{aligned}$$

この式は、サッカー情報配信サービスから届くすべての配信単位の *ITS* 属性の値が、直前の水曜日の午前 0 時を表す値以上かつその翌日の午前 0 時を表す値未満であることを明示したものである。

一方、天気予報配信サービスでは、配信単位が毎日午前 6 時に届くことを仮定した。よって、以下のように性質を記述できる。

$$\begin{aligned}
&Weather \equiv\\
&\quad \sigma_{Weather.ITS=after_{0:6:0:0}(}\\
&\qquad previous_{*:0:0:0}(Weather.ITS))(Weather)
\end{aligned}$$

この式は、すべての天気予報の *ITS* 属性の値が午前 6 時に等しいことを明示している。

なお、本稿では配信型情報源を含めて ITS 以外の属性値に関する制約は考慮しない (特別の制約は存在しない) ものとする。

5 配信要求記述の正当性

与えられた配信要求記述から ECA ルールを生成する前に、配信要求の正当性を検証することが望ましい。ここでは、正当性の基準として整合性と充足可能性の概念について説明する。これらの性質の具体的検証方法については 6 節で述べる。

5.1 整合性

ユーザが、以下のような要求を記述したとする。

$$O_{new} = \Omega_{immediate(Soccer.ITS)}(Soccer \times Weather)$$

これは「サッカー情報が到着したら、そのデータに過去および未来すべての天気予報を組み合わせて直ちに配信して欲しい」という要求記述となる。サッカー情報が到着した時点では、せいぜい 4 週間先の天気予報があるに過ぎず、それ以上先の天気予報はまだ到着していない。当然、システムは要求された内容を満たす統合結果を生成不可能なため、この要求は実現できない。このような実現不可能な要求を検出するために整合性の概念を導入する。

整合性の概念について触れる前に、タイムスタンプ関数 f の逆関数 f^{-1} について説明する。逆関数の定義は以下の通りである。

$$f^{-1}(t) = \{u | f(u) = t\}.$$

3.3 節で導入した各タイムスタンプ関数に対する逆関数値は以下のような式で計算できる。

1. $\mathbf{immediate^{-1}}(t) = \{u | u = t\}$
2. $\mathbf{next_p^{-1}}(t) = \{u | previous_p(t) \leq u < t\}$
3. $\mathbf{previous_p^{-1}}(t) = \{u | t < u \leq next_p(t)\}$
4. $\mathbf{after_{\delta t}^{-1}}(t) = \{u | u = before_{\delta t}(t)\}$
5. $\mathbf{before_{\delta t}^{-1}}(t) = \{u | u = after_{\delta t}(t)\}$

一般に $t = f(I_k.ITS)$ のとき、$u \in f^{-1}(t)$ を考える際には、I_k の性質記述 $\sigma_{PROP_k}(I_k)$ における条件式 $PROP_k$ を満たす u のみを考えればよい。以下では、条件 $PROP_k$ 中に出現する $I_k.ITS$ を u で置き換えた条件式を $PROP_k(u)$ と表す。

定義 1 配信要求記述 $O_{new} = \Omega_{f(I_k.ITS)}(E(I_1, \cdots, I_n))$ と各配信型情報源の性質記述 $I_1 \equiv \sigma_{PROP_1}(I_1), \cdots, I_n \equiv \sigma_{PROP_n}(I_n)$ が与えられたとする。このとき、E' を以下の式とする。

$$\begin{aligned}
&E'(I_1, \cdots, I_n, t) =\\
&\qquad \sigma_{I_k.ITS \in f^{-1}(t)}(E(\sigma_{PROP_1}(I_1), \cdots, \sigma_{PROP_n}(I_n)))
\end{aligned}$$

このとき、$PROP_k(u)$ を満たす任意の時刻 u について以下の等式が成立するとき、その配信要求記述は**整合**しているという。

$$\begin{aligned}
&E'(I_1, \cdots, I_n, f(u)) =\\
&\qquad E'(\sigma_{ITS \leq f(u)}(I_1), \cdots, \sigma_{ITS \leq f(u)}(I_n), f(u))
\end{aligned}$$

□

E' は、与えられた配信型情報源の性質記述を考慮した上で、時刻 t において配信すべきタプルを選択する式である。定義 1 は、具体的な配信予定時刻として指定され得る時刻 $t = f(u)$ に対して、その時点で要求される統合結果が、その時刻以前に届いた情報のみから生成可能でなければならないことを意味している。

6 節に示す検証手順では以下の定理を用いる。

定理 1 全ての情報源 I_i $(i = 1,\ldots,n)$ に対して、以下の条件が成り立つことが、E' が整合しているために必要十分である。

$$E'(I_1,\ldots,\sigma_{ITS_i>f(u)}(I_i),\ldots,I_n,f(u)) = \phi$$

□

証明 1 （必要性の証明） 定義 *1* の式から定理 *1* の式を導出する。

$$\begin{aligned}
&E'(I_1,\ldots,\sigma_{ITS_i>f(u)}(I_i),\ldots,I_n,f(u))\\
&= \sigma_{ITS_i>f(u)}(E'(I_1,\ldots,I_n,f(u)))\\
&\quad (3.3\text{ の }E\text{ に関する仮定})\\
&= \sigma_{ITS_i>f(u)}(\\
&\qquad E'(\sigma_{ITS_1\le f(u)}(I_1),\ldots,\sigma_{ITS_n\le f(u)}(I_n),f(u)))\\
&\qquad (\text{定義 } 1)\\
&= E'(\sigma_{ITS_1\le f(u)}(I_1),\ldots,\sigma_{ITS_i>f(u)\wedge ITS_i\le f(u)}(I_i),\ldots\\
&\qquad \ldots,\sigma_{ITS_n\le f(u)}(I_n),f(u))\\
&\qquad (3.3\text{ の }E\text{ に関する仮定})\\
&= E'(\sigma_{ITS_1\le f(u)}(I_1),\ldots,\sigma_{false}(I_i),\ldots\\
&\qquad \ldots,\sigma_{ITS_n\le f(u)}(I_n),f(u))\\
&= \phi \quad (3.3\text{ の }E\text{ に関する仮定})
\end{aligned}$$

（十分性の証明） 定理 *1* の式から定義 *1* の式を導く。

$$\begin{aligned}
&E'(I_1,\ldots,I_n,f(u))\\
&= \sigma_{true}(E'(I_1,\ldots,I_n,f(u)))\\
&= \sigma_{\bigwedge_i(ITS_i\le f(u)\vee ITS_i>f(u))}(E'(I_1,\ldots,I_n,f(u)))\\
&= \sigma_{\bigwedge_i(ITS_i\le f(u))\vee\bigvee_i(ITS_i>f(u))}(E'(I_1,\ldots,I_n,f(u)))\\
&= \sigma_{\bigwedge_i ITS_i\le f(u)}(E'(I_1,\ldots,I_n,f(u)))\\
&\qquad \cup\bigcup_i\sigma_{(ITS_i>f(u))}(E'(I_1,\ldots,I_n,f(u)))\\
&= E'(\sigma_{ITS_1\le f(u)}(I_1),\ldots,\sigma_{ITS_n\le f(u)}(I_n),f(u))\\
&\qquad \cup\bigcup_i E'(I_1,\ldots,\sigma_{ITS_i>f(u)}(I_i),\ldots,I_n,f(u))\\
&\qquad (3.3\text{ の }E\text{ に関する仮定})\\
&= E'(\sigma_{ITS_1\le f(u)}(I_1),\ldots,\sigma_{ITS_n\le f(u)}(I_n),f(u))\\
&\qquad (E'(I_1,\ldots,\sigma_{ITS_i>f(u)}I_i,\ldots,I_n,f(u)) = \phi)
\end{aligned}$$

□

5.2 充足可能性

ユーザが以下のような配信要求を記述したとする。

$$O_{new} = \Omega_{next_{Tue:0:0:0}(Soccer.ITS)}(\\ \sigma_{Soccer.ITS\ge previous_{Sun:0:0:0}(Soccer.ITS)}(Soccer))$$

これは「毎週火曜日の午前 0 時に、その週の日曜の午前 0 時以降に届いたサッカー情報を配信して欲しい」という意味になる。しかしサッカー情報は毎週水曜日にならないと届かないのであるから、火曜日の午前 0 時の時点でこの要求を処理しようとしても配信できる情報が存在せず、ユーザの手元に情報が届くことはない。

このような要求に対して ECA ルールを生成することは無駄であるし、ユーザがそのような要求を意図して記述することは考えにくく、記述ミスの可能性が高い。よって、常に要求された統合結果が空になり、将来に渡って配信が行われないような要求は前もって検出する必要がある。この点に関する性質として、充足可能性の概念を導入する。

定義 2 配信要求記述 $O_{new} = \Omega_{f(I_k.ITS)}(E(I_1,\cdots,I_n))$ と各配信型情報源の性質記述 $I_1 \equiv \sigma_{PROP_1}(I_1),\cdots,I_n \equiv \sigma_{PROP_n}(I_n)$ が与えられたとする。このとき、E' を以下の式とする。

$$E'(I_1,\cdots,I_n,t) =\\ \sigma_{I_k.ITS\in f^{-1}(t)}(E(\sigma_{PROP_1}(I_1),\cdots,\sigma_{PROP_n}(I_n)))$$

このとき、現在時刻 *now* 以降のある時刻 $f(u)$（ただし、u は $PROP_k(u)$ を満たす）において以下の等式が成立するとき、その配信要求記述は**充足可能**であるという。

$$E'(I_1,\cdots,I_n,f(u)) \neq \phi$$

□

定義 2 は、充足可能な配信要求記述により定義された配信サービスからは、将来いずれかの時点で情報が配信されることを表している。

6 正当性の検証

本節では、5 節で説明した整合性と充足可能性を実際に検証する手法について述べる。

6.1 条件式の変換

時刻に関する条件式はタイムスタンプ関数を含んでおり、その扱いに工夫が必要である。本研究では時刻 t をある起点からの経過時間を最小時間単位を用いて表した非負の整数値 n で表現するものとし、タイムスタンプ関数を含んだ時刻に関する条件式を整数値をとる変数に関する条件式に変換することで検証を可能とする。最小時間単位としては、1 秒を整数として表現できるものであれば以下の議論ではどのようなものでも良い。また、起点は過去のある日曜日の午前 0 時とするが、他の起点を選択した場合でも本質的な変更点はない。時刻に関する条件式を整数の条件式に変換するためには 3.3 節で示したタイムスタンプ関数を整数上の関数に変換する必要がある。タイムスタンプ関数の変換規則は以下のように定義できる。

$$\begin{aligned}
immediate(n) &\Rightarrow n\\
next_p(n) &\Rightarrow am+b\\
&\quad (\text{ただし } n < am+b \le n+a)\\
previous_p(n) &\Rightarrow am+b\\
&\quad (\text{ただし } n-a \le am+b < n)\\
after_{\delta t}(n) &\Rightarrow n+c\\
before_{\delta t}(n) &\Rightarrow n-c
\end{aligned}$$

ただし、a,b,c は関数のパラメータ $p,\delta t$ に応じて決まる整数の正定数である。

immediate, after, before の変換は単純である。ここでは *next*, *previous* についてより詳しく説明する。$next_p(n)$, $previous_p(n)$ は、パラメータ p で指定された条件にマッチする時刻の中でもっとも n に近い時刻を返す関数である。このときパラメータ p にマッチする時刻の集合は、一定の周期を持った時刻の集合になる。たとえば $p =_{*:6:0:0}$ にマッチする時刻の集合は、毎日の午前 6 時である。これは次のよう記述できる。

$$
\begin{array}{l}
\sigma_{previous_{*:0:0:0}(t)\leq Soccer.ITS \wedge Soccer.ITS<t}\ (\\
\quad (\\
\quad \sigma_{previous_{Wed:0:0:0}(Soccer.ITS)\leq Soccer.ITS}\\
\quad\quad \wedge Soccer.ITS<after_{1:0:0:0}(previous_{Wed:0:0:0}(Soccer.ITS))\\
\quad\quad\quad (Soccer)\\
\quad)\\
\quad \bowtie_{Soccer.date=Weather.date}\\
\quad\quad \wedge previous_{*:0:0:0}(next_{*:0:0:0}(Soccer.ITS))\leq Weather.ITS\\
\quad\quad \wedge Weather.ITS<next_{*:0:0:0}(Soccer.ITS)\\
\quad (\\
\quad \sigma_{Weather.type=\text{`7 days'}}\\
\quad\quad \wedge Weather.ITS=after_{0:6:0:0}(previous_{*:0:0:0}(Weather.ITS))\\
\quad\quad\quad (Weather)\\
\quad)\\
\quad \bowtie_{Weather.location=Stadium.location}\\
\quad\quad \wedge Soccer.place=Stadium.name\\
\quad Stadium\\
)
\end{array}
$$

図 3: 統合例における E'

$$\{u|u = day\cdot m + 6\,hour\}$$

ただし、*day*,*hour* は 1 日の長さと 1 時間の長さを最小時間単位を用いて表す定数とする。

これを一般化し、パラメータ p にマッチする時刻の集合を以下のように整数の集合として表す。

$$\{u|am + b\}$$

a はパラメータ p に含まれている'*' の位置によって決まり、b は記述されている数値によって決まる。さらに、*previous*,*next* は p にマッチする集合のうち、与えられた時刻に最も近い時刻を返すので、それを表現するために以下の条件を付け加える。

$$\begin{array}{rcccl} n & < & next_p(n) & \leq & n+a \\ n-a & \leq & previous_p(n) & < & n \end{array}$$

例えば、$previous_{*:6:0:0}(n)$ であれば以下のようになる。

$$day\cdot m + 6\,hour \text{（ただし　} n - day \leq day\cdot m + 6\,hour < n)$$

以下では、3.3 節で示した配信要求記述と 4 節で示した情報源の性質記述について、条件式の変換の例を示す。まず、配信要求記述を 5 節で示した $E'(I_1,\ldots,I_n,t)$ に変形する (図 3)。ここから時刻に関する比較条件を抜き出し、t に対して $f(u)$（この場合は $next_{*:0:0:0}(u)$）を代入すると以下のようになる。

$$\begin{array}{l}
previous_{*:0:0:0}(next_{*:0:0:0}(u)) \leq Soccer.ITS\\
Soccer.ITS < next_{*:0:0:0}(u)\\
previous_{Wed:0:0:0}(Soccer.ITS) \leq Soccer.ITS\\
Soccer.ITS < after_{1:0:0:0}(previous_{Wed:0:0:0}(Soccer.ITS))\\
previous_{*:0:0:0}(next_{*:0:0:0}(Soccer.ITS)) \leq Weather.ITS\\
Weather.ITS < next_{*:0:0:0}(Soccer.ITS)\\
Weather.ITS =\\
\quad after_{0:6:0:0}(previous_{*:0:0:0}(Weather.ITS))
\end{array}$$

また、u は $PROP_{Soccer}(u)$ を満たすものでなければならない。よって次の条件式も追加される。

$$\begin{array}{l}
previous_{Wed:0:0:0}(u) \leq u\\
u < after_{1:0:0:0}(previous_{Wed:0:0:0}(u))
\end{array}$$

これらの式に対して前述の変換規則を適用する。上記の各項は以下の右辺のような式に変換される。ただし、$x_1,\ldots,x_{10}$ は非負の整数値をとる変数である。

$$\begin{array}{l}
u \Rightarrow x_1\\
next_{*:0:0:0}(u) \Rightarrow day\cdot x_2\\
\quad (\text{ただし } x_1 < day\cdot x_2 \leq x_1 + day)\\
previous_{*:0:0:0}(next_{*:0:0:0}(u)) \Rightarrow day\cdot x_3\\
\quad (\text{ただし } day\cdot x_2 - day \leq day\cdot x_3 < day\cdot x_2)\\
Soccer.ITS \Rightarrow x_4\\
previous_{Wed:0:0:0}(Soccer.ITS) \Rightarrow week\cdot x_5 + 3day\\
\quad (\text{ただし } x_4 - week \leq week\cdot x_5 + 3day < x_4)\\
after_{1:0:0:0}(previous_{Wed:0:0:0}(Soccer.ITS))\\
\quad \Rightarrow week\cdot x_5 + 4day\\
next_{*:0:0:0}(Soccer.ITS) \Rightarrow day\cdot x_6\\
\quad (\text{ただし } x_4 < day\cdot x_6 \leq x_4 + day)\\
previous_{*:0:0:0}(next_{*:0:0:0}(Soccer.ITS)) \Rightarrow day\cdot x_7\\
\quad (\text{ただし } day\cdot x_6 - day \leq day\cdot x_7 < day\cdot x_6)\\
Weather.ITS \Rightarrow x_8\\
previous_{*:0:0:0}(Weather.ITS) \Rightarrow day\cdot x_9\\
\quad (\text{ただし } x_8 - day \leq day\cdot x_9 < x_8)\\
after_{0:6:0:0}(previous_{*:0:0:0}(Weather.ITS))\\
\quad \Rightarrow day\cdot x_9 + 6hour\\
previous_{Wed:0:0:0}(u) \Rightarrow week\cdot x_{10} + 3day\\
\quad (\text{ただし } x_1 - week \leq week\cdot x_{10} + 3day < x_1)\\
after_{1:0:0:0}(previous_{Wed:0:0:0}(u))\\
\quad \Rightarrow week\cdot x_{10} + 4day
\end{array}$$

これらを上記の時刻に関する条件式に代入すると以下の条件式群が得られる。

$$\begin{array}{ll}
x_1 < day\cdot x_2 & day\cdot x_6 - day \leq day\cdot x_7\\
day\cdot x_2 \leq x_1 + day & day\cdot x_7 < day\cdot x_6\\
day\cdot x_2 - day \leq day\cdot x_3 & day\cdot x_7 \leq x_8\\
day\cdot x_3 < day\cdot x_2 & x_8 \leq day\cdot x_6\\
day\cdot x_3 \leq x_4 & x_8 - day \leq day\cdot x_9\\
x_4 < day\cdot x_2 & day\cdot x_9 < x_8\\
x_4 - week \leq week\cdot x_5 + 3day & x_8 = day\cdot x_9 + 6hour\\
week\cdot x_5 + 3day < x_4 & x_1 - week \leq week\cdot x_{10} + 3day\\
week\cdot x_5 + 3day \leq x_4 & week\cdot x_{10} + 3day < x_1\\
x_4 < week\cdot x_5 + 4day & week\cdot x_{10} + 3day \leq x_1\\
x_4 < day\cdot x_6 & x_1 < week\cdot x_{10} + 4day\\
day\cdot x_6 \leq x_4 + day &
\end{array}$$

6.2 条件式の検証

前節の変換作業により、時刻に関する条件式を以下のような線形不等式で表される条件の集合 (論理積) に変換することができる (実際にはさらに $=$ や $<$ を $\leq$ へ変換する必要がある)。

$$\begin{array}{c}
a_{11}x_1 \quad +\ldots+ \quad a_{1n}x_n \quad \leq b_1\\
\vdots\\
a_{m1}x_1 \quad +\ldots+ \quad a_{mn}x_n \quad \leq b_m\\
\text{ただし } x_1,\ldots,x_n \geq 0
\end{array}$$

このような変換を行ったとき、整合性の検証とは、定理 1 から、上記の線形不等式へさらに条件 $ITS_i > f(u)$ を加えたものに $x_1,\ldots,x_n$ の非負の整数解が存在するかどうかを調べることと等価である。また、充足可能性の検証は、定義 2 より、上記の線形不等式へ条件 $f(u) \geq now$ を追加したものに非負の整数解が存在するかどうかを調べることと等価である。

線形不等式集合に解が存在するかどうかを判定する問題は、線形計画問題と深く関係している。一般に線形計画問題とは、非負の変数 $x_1 \ldots x_n$ に対する線形不等式制約が与えられたとき、その制約の範囲内で目的関数を最大化するような解を探す問題である。

上記線形不等式においてすべての b_i が $b_i \geq 0$ であるならば、その制約を充足する自明な解 $x_1 = \ldots = x_n = 0$ が存在する。$b_i < 0$ となる b_i が含まれる場合に制約式を充足する解が存在するかを判定するには、以下の線形計画問題を解けばよいことが知られている [16, 17]。

$$
\begin{array}{ll}
\text{最大化：} & -x_a \\
\text{制約式：} & a_{11}x_1 + \ldots + a_{1n}x_n \quad \leq x_a + b_1 \\
& \vdots \\
& a_{m1}x_1 + \ldots + a_{mn}x_n \quad \leq x_a + b_m \\
& x_1, \ldots, x_n, x_a \geq 0
\end{array}
$$

この線形計画問題には、制約を充足する解 $x_1 = \ldots = x_n = 0$, $x_a = -min_i(b_i)$ が存在している。しかし、この問題の解が元の制約式の解となるための必要十分条件は $x_a = 0$ となることである。よって目的関数 $-x_a$ を最大化し、その値を 0 にできならなければならない。目的関数の最大化は、シンプレックス法 [16, 17] などのアルゴリズムを適用することで行うことができる。目的関数が 0 のときの解が元の制約式を満たす解となる。目的関数の値を 0 に最大化できないときは、元の制約式を充足する解が存在しないことを表す。

ただし、我々が変換した条件式の変数はすべて整数制約を持つので、実際は整数線形計画問題として整数解を求める必要がある。シンプレックス法を整数制約に適用した方法 [16, 17] もすでに知られているので、それを用いることで整数制約に対しても同様に判定を行うことができる。整数線形計画問題は、NP 完全であることが知られており、問題サイズの指数オーダの時間がかかる。しかし配信要求記述の検証では、変数の数は統合対象となる情報源の数と条件式に記述された *next*, *previous* の数によって決定されるため、変数の数が膨大になることはなく、十分実用的な時間で解くことができる。

6.3 検証手順

以上の議論をまとめると、配信要求記述の検証の手順は以下のようになる。

1. 配信要求記述 $E(I_1, \ldots, I_n)$ と各情報源の性質記述から、$E'(\sigma_{PROP_1}(I_1), \ldots, \sigma_{PROP_i}(I_n), t)$ を導出し、時刻に関する条件式を抽出する。
2. t を $f(u)$ で置き換え、$PROP_k(u)$ を条件式に追加する。
3. 6.1 に述べた方法で整数に関する線形不等式集合を得る。
4. 整合性の判定
   (a) 各情報源 I_i ごとに以下の処理を繰り返す。全ての i に対して結果が真ならば配信要求記述は整合性を満たす。そうでなければ、整合性を満たさない。
   (b) 元の条件式に $ITS_i > f(u)$ から得られる条件を付け加え、整数線形計画問題を導出する。
   (c) 上記問題を解いて、目的関数が 0 に最大化可能であれば整合性を満たさない解が存在することになるので偽、そうでなければ真とする。
5. 充足可能性の判定
   (a) 元の条件式に $f(u) \geq now$ から得られる条件を加え、整数線形計画問題を導出する。
   (b) 上記問題を解いて、目的関数が 0 に最大化可能であるならば配信要求記述は充足可能性を満たす。そうでなければ充足可能性を満たさない。

7 ECA ルールの生成

本稿では ECA ルール生成の概要を述べる。より詳細については [13, 14] を参照されたい。ECA ルールの生成は、以下の手順に基づいて行われる。

配信要求記述 $O_{new} = \Omega_{f(I_k.ITS)}(E(I_1, \cdots, I_n))$ および各情報源についての性質記述 $I_1 \equiv \sigma_{PROP_1}(I_1), \cdots, I_n \equiv \sigma_{PROP_n}(I_n)$, が与えられたとする。

1. 配信要求記述から以下のような E' を導出する。
   $E'(I_1, \cdots, I_n, t) =$
   $\sigma_{I_k.ITS \in f^{-1}(t)}(E(\sigma_{PROP_1}(I_1), \cdots, \sigma_{PROP_n}(I_n)))$
2. 6 節で示した手法により、E' の整合性と充足可能性を検証する。
3. 従来のリレーショナル代数における最適化手法 [18] と同様の方法で選択条件をプッシュダウンし、以下の形に書き換える。
   $$E''(\sigma_{P_1(t) \wedge Q_1}(I_1), \cdots, \sigma_{P_n(t) \wedge Q_n}(I_n), t),$$
   ただし、$P_i(t)$ は情報源 I_i の ITS 属性に関する選択条件で、Q_i はそれ以外の属性に関する選択条件である。
4. 各 I-sequence リレーションごとに蓄積ルールを導出する。
5. 生成ルールを導出する。
6. 各 I-sequence リレーションごとに廃棄ルールを導出する。

8 関連研究

DBIS[19, 20] と *Muffin*[21] は配信型情報源から情報の抽出することを目的とするシステムである。DBIS は複数の情報源にアクセスし、ユーザプロファイルに基づく情報の選択が行える。Muffin はユーザプロファイルに基づいて新たな仮想チャネルを作ることができ、複数の配信型情報源からの情報の選択には、配信の頻度や新鮮度、人気度を用いている。これらの研究は配信型情報源から情報を抽出することを目的としたシステムである。

OpenCQ[22] は分散情報源に対する統合を行うためのシステムである。このシステムは、イベント駆動型の処理に continual query を採用している。continual query は問合せ、発火条件、終了条件の 3 つから構成されている。更新イベントが発生したときに発火条件が真であると、問合せが実行される。この研究はイベント駆動による情報統合という点で我々の研究と関連がある。しかし OpenCQ では、continual query の自動生成は考えられていないので、ユーザが直接記述する必要がある。

他にも配信型情報源の統合に関連した研究はあるが、代数式による配信要求記述から ECA ルールを生成するアプローチは我々のオリジナルである。また、本研究は配信型情報源から届く情報の到着時刻の性質を考慮するように拡張した枠組みである。

周期的な時間データ列を扱う関連研究として、以下のようなものがある。

- 論文 [23] は *generalized relation* とそれらに対する代数を提案している。generalized relation では、*linear repeating point*(LRP) という $kn+c$ (k,c は定数、n は任意の整数) で表される整数の列を扱うことができ、周期的な値をとる属性は LRP によって一般化して表現される。しかし、周期的でない属性はそのまま通常のリレーションとして表されるので、コンテンツが配信されるまでわからない配信型情報源には、generalized relation は適さない。
- 論文 [24] は *strong periodicity* と *near periodicitiy* の 2 種類の周期性について述べている。strong periodicity は、ある事象が一定の間隔で発生する場合であり、LRP によって表現される。一方 near periodicity は、事象が一定の周期で発生するが発生間隔は必ずしも一定でない場合であり、LRP の周辺にある点として表現される。[24] では、これらの概念に基づいて問合せ言語の表現能力の比較を行っている。我々の研究は、strong periodicity を考慮している。
- 論文 [25] は、異なった周期を持つ複数のデータ列から、新鮮度と同期度の概念に基づいて適切なデータの組合せを選択する手法を提案している。また、連続問い合わせのために最適なデータの組合せをあらかじめ計算する手法を示している。我々の研究においては、配信の周期は必ずしも固定ではなく、統合要求は明示的に代数によって指定を行っている。

9 まとめと今後の課題

本稿では、リレーショナル代数式による配信要求記述から ECA ルールを生成し、新たな情報配信サービスを定義できる配信型情報源統合環境において、配信要求記述の正当性の検証を行うための手法について述べた。配信要求が持つべき性質として整合性と充足可能性を導入し、それらを検証するための手法として、時刻に関する条件式を整数の条件式に変換して、線形計画問題の解法を利用するアプローチを示した。

本稿で提案した枠組みは、現在 3 節で示したアーキテクチャをもとにして、我々の研究グループが研究開発を進めてきたプロトタイプシステム上で実装作業を行なっている。今後の課題としては、プロトタイプシステムを用いた検証手法の評価と、より一般的な情報源の持つ性質への対応などが挙げられる。

謝辞

本研究の一部は、日本学術振興会科学研究費基盤研究 (B)(12480067)、奨励研究 (A)(12780183)、および文部科学省科学研究費特定領域研究 (C)(13224008) による。

参考文献

[1] R. Domenig and K. R. Dittrich. An Overview and Classification of Mediated Query Systems. *ACM SIGMOD Record*, 28(3), pp.63–72, 1999.

[2] A. Elmagarmid, M. Rusinkiewicz, and A. Sheth (Eds.). Management of Heterogeneous and Autonomous Database Systems. Morgan Kaufmann, 1999.

[3] H. Kitagawa, A. Morishima, and H. Mizuguchi. Integration of Heterogeneous Information Sources in InfoWeaver. *Advances in Database and Multimedia for the New Century – A Swiss/Japanese Perspective*, World Scientific Publishing, pp. 124–137, 2000.

[4] A. Morishima and H. Kitagawa. InfoWeaver: Dynamic and Tailor-Made Integration of Structured Documents, Web, and Databases. *Proc. ACM DL '99*, pp. 235–236, 1999.

[5] G. Wiederhold. Mediators in the Architecture of Future Information Systems. *IEEE Computer*, Vol 25, No 3, pp.38–49, 1992.

[6] M. Franklin and S. Zdonik. Data in Your Face: Push Technology in Perspective. *Proc. ACM SIGMOD*, pp.516–519, Jun, 1998.

[7] S. Ramakrishnan and V. Dayal. The PointCast Network. *ACM SIGMOD Record*, 27(2), p. 520, 1998.

[8] Infogate Inc. Infogate. http://www.infogate.com/.

[9] TV-Asahi Data Inc. ADAMS. http://www.tv-asahidata.com/.

[10] INFOCITY Inc. bitcast. http://www.bitcast.ne.jp/.

[11] N. W. Paton and O. Diaz. Active Database Systems. *ACM Comp. Serv.*, 31(1), 1999.

[12] H. Mizuguchi, H. Kitagawa, Y. Ishikawa, and A. Morishima. A Rule-oriented Architecture to Incorporate Dissemination-based Information Delivery into Information Integration Environments. *Proc. 2000 ADBIS-DASFAA*, pp. 185–199, 2000.

[13] H. Kitagawa, T. Kajino, and Y. Ishikawa. Algebraic Service Specification and Rule Generation for Integrating Multiple Dissemination-Based Information Sources. *Proc. 7th International Conference on Database Systems for Advanced Applications*, pp. 344–351, 2001.

[14] Y. Watanabe, H. Kitagawa, and Y. Ishikawa, Integration of Multiple Dissemination-Based Information Sources Using Source Data Arrival Properties. *Proc. 2nd International Conference on Web Information System Engineering*, 2001.

[15] H. Garcia-Molina, W. Labio, and J. Yang. Expiring Data in a Warehouse. *Proc. 24th VLDB Conference*, pp. 500–511, 1998.

[16] H. A. Eiselt, and C.-L. Sandblom. Integer Programming and Network Models. Springer, 2000.

[17] R. S. Garfinkel, and G. L. Nemhauser. INTEGER PROGRAMMING. John Wiley & Sons, Inc. 1972.

[18] P. G. Selinger, M. M. Astrahan, D. D. Chamberlin, R. A. Lorie, and T. G. Price. Access Path Selection in a Relational Database Management System. *Proc. ACM SIGMOD Conference*, pp. 23–34, 1979.

[19] D. Aksoy, M. Altinel, R. Bose, U. Cetintemel, M. Franklin, J. Wang, and S. Zdonik. Research in Data Broadcast and Dissemination. *Proc. AMCP '98*, pp. 194–207, 1998.

[20] M. Altinel, D. Aksoy, T. Baby, M. Franklin, W. Shapiro, and S. Zdonik. DBIS Toolkit – Adaptable Middleware for Large-Scale Data Delivery. *Proc. ACM SIGMOD '99*, 1999.

[21] M. Qiang, H. Kondo, K. Sumiya, and K. Tanaka. Virtual TV Channel: Filtering Merging and Presenting Internet Broadcasting Channels. *ACM DL Workshop on WOWS*, 1999.

[22] L. Liu, C. Pu, and W. Tang. Continual Queries for Internet Scale Event-Driven Information Delivery. *IEEE TKDE*, 11(4), pp. 610–628, 1999.

[23] F. Kabanza, J. M. Stevenne, and P. Wolper. Handling Infinite Temporal Data. *Proc. 9th ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems*, pp. 392–403, 1990.

[24] A. Tuzhilin, and J. Clifford. On Periodicity in Temporal Databases. *Information System*, 20(8), pp.619–639, 1995.

[25] K. Munakata, M. Yoshikawa, and S. Uemura. Integrating Periodic Data Sequences Based on Freshness and Synchronousness. *Proc. 10th IEEE Workshop on Research Issues in Data Engineering*, pp. 47–54, 2000.