

## Sharing Common Operations to Integrate Dissemination-based Information Sources

渡辺 陽介<sup>†</sup>, 北川 博之<sup>††</sup>, 石川 佳治<sup>††</sup>

Yousuke Watanabe, Hiroyuki Kitagawa, Yoshiharu Ishikawa

## 1 はじめに

近年のネットワーク技術の発達により、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いる [1]。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 [2] が現在注目されている。

我々の研究グループは、ECA ルール [3] を用いて能動的な統合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 [4]。しかし、必要な ECA ルールをユーザがルール間の整合性を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我々は、ユーザが与えた代数的配信要求記述から ECA ルールを自動的に生成する方式を提案した。

本稿ではネットワーク上の大勢のユーザから利用されることを想定した配信型情報源統合環境について述べる。同じ情報源に対する配信要求が大量に与えられた場合、配信要求内に共通の演算を含んだものが多くなるので、問合せの中間結果を共有する手法が有効である。演算子の共有に関しては、これまで複数問合せに対する最適化手法 [6] が提案されてきている。しかし、我々の統合環境では配信要求記述に配信スケジュールがあるため、処理内容は同じでも配信スケジュールが異なるような配信要求が存在する。本研究では、時間要件を考慮した演算子の共有とそれを用いた ECA ルールの生成を行う。

## 2 統合利用例

まず、統合対象として次の3つの情報源を仮定する。

- 株価情報サービス (Quote): 毎日の株価の終値を企業ごとに配信する配信型情報源で、サーバから配信される情報には企業名 (name)、終値 (price) が含まれるものとする。また、情報が配信されるのは毎日の15時とする。
- 企業ニュース情報サービス (News): 各企業に関するニュースを配信する配信型情報源で、企業名 (name)、見出し (head)、コンテンツ (content) が含まれている。ニュースが配信されるのは9時から17時までとする。
- 企業リレーション (Company): 各企業に関する情報を格納した通常のリレーショナルデータベース。企業名 (name)、資本金 (capital)、従業員数 (employee) などの情報が含まれる。

なお、ここではすべての情報源はリレーションで表現されるものとする。また配信型情報源 (Quote, News) は、送られてきた配信単位の到着時刻を表す ITS 属性を持つものとする。

これら3つの情報源から新たな情報配信サービスを定義する要求として次の2つを考える。

1. A社の株価情報が400より大きければ、その日のニュース記事と企業情報をつけて18時に配信して欲しい。
2. A社の株価情報が300より大きければ、その日のニュース記事と企業情報をつけて22時に配信して欲しい。

[4]の枠組に従うと、これらの配信要求は図1、図2のように与えることができる。ただし、 $\sigma, \wedge$  はリレーショナル代数の選択演算と結合演算、 $\Omega$  は配信予定時刻を指定する演算であるとする。また、*next*, *previous* はタイムスタンプ関数である。これらについては [4] で詳しく述べられている。

我々の統合環境は、図1や図2の配信要求から ECA ルールを導出し、情報の到着や配信スケジュールに応じたイベント駆動的な蓄積・統合・配信の処理を実現する。

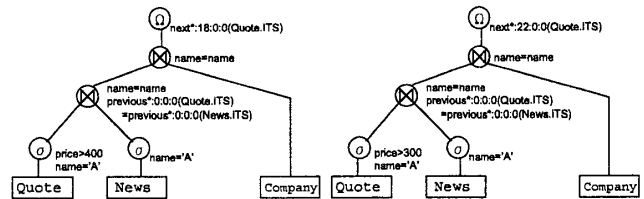


図1: 配信要求1

図2: 配信要求2

## 3 配信特性

本統合環境では、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情報を配信特性 [5] として記述する。2節の株式情報配信サービスでは情報が毎日15時に届くと仮定した。この配信特性は、以下のように記述することができる。

$$Quote \equiv \sigma_{Quote.ITS=after_{0:15:0:0}(previous_{*,0:0:0}(Quote.ITS))} (Quote)$$

この式は、株式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、15時であることを表している。

一方、企業ニュース配信サービスでは以下のように記述できる。

$$News \equiv \sigma_{after_{0:9:0:0}(previous_{*,0:0:0}(News.ITS)) \leq News.ITS \wedge News.ITS < after_{0:17:0:0}(previous_{*,0:0:0}(News.ITS))} (News)$$

この式は、すべてのニュースの ITS 属性の値が9時から17時の範囲に含まれていることを明示している。

## 4 演算の共有

演算の共有については [6] などの既存研究で考察がされているが、共有する際に配信要求の時間要件を考慮しなければならぬ点が大きく異なる。

## 4.1 選択演算のグループ化

本節では選択演算のグループ化と評価結果の共有について述べる。まず配信型情報源のリレーションに新たに tag 属性を用意し、各タプルがどの配信要求から必要とされているかを記録する。tag 属性はビットの配列で、各ビットがそれぞれの配信要求に対応しているものとし、到着したばかりの情報の tag 属性の各ビットにはすべての1が立っているとする。

次に選択演算をグループ化するために Tag 演算を導入する。Tag 演算はタプルと選択条件の集合を受け取り、各条件を評価した結果をタプルの tag 属性に書き込む。その配信要求にとって不要なタプルであれば tag 属性の対応するビットはクリアされる。また、Tag 演算の戻り値は、タプルがいずれかの配信要求で必要とされる場合に真とする。

ユーザへの配信処理を行う前に、tag 属性を調べてタプルが配信要求にとって本当に必要かどうか判定する処理を行わなければならない。ある配信要求がそのタプルを必要とするかどうかを判定するための関数として isNeeded(id, attribute) を用いる。ここで、id は配信要求の番号、attribute は tag 属性の名前とする。

統合例における図1と図2は、それぞれ Quote と News に対する選択演算を行っており、これらは図3のようにまとめることができる。

## 4.2 統合結果の共有

複数の配信要求が与えられた場合に統合結果を共有するためには、配信要求の時間要件を考慮する必要がある。

<sup>†</sup> 筑波大学 システム情報工学研究科

<sup>††</sup> 筑波大学 電子・情報工学系

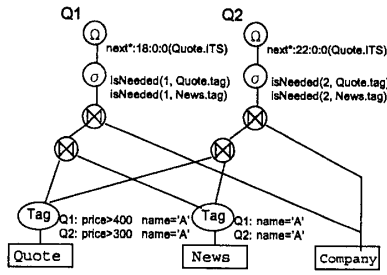


図 3: 選択演算のグループ化

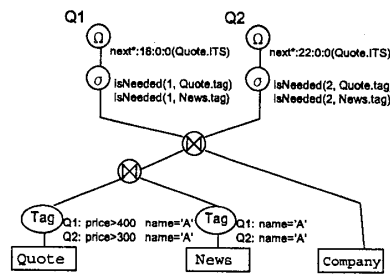


図 4: 統合結果の共有

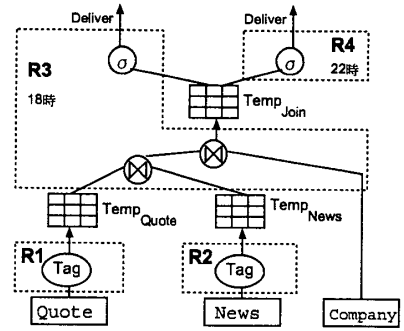


図 5: 複数要求に対するECAルール

図 3 において配信要求 1 と配信要求 2 を調べると、結合演算の部分が共通しているが、これらは配信スケジュールが違っている。生成する時間によって統合結果が変わる場合は、この 2 つを同時に実行し、結果を共有することはできない。しかし、株式情報配信サービスは毎日 15 時のみに配信が行われ、またニュース情報配信サービスは毎日 17 時でその日の配信が終了するため、18 時と 22 時のどちらの時間に統合結果を生成しても中身は変わらない。この性質は 3 節の配信特性を用いることによって保証される。よって、この場合は 18 時に配信要求 1 で生成した統合結果を配信要求 2 においても使うことができ、図 3 の結合演算を共有することができる (図 4)。

より一般的には、配信予定時刻と結合条件と配信特性から統合結果の生成に必要な配信単位の時間範囲を求め、それらが共有したい配信要求同士で同一であるかを調べることによって共有できるかを判定する。

## 5 ECA ルールの導出例

演算を共有した配信要求から ECA ルールを生成するときの方針は以下のようになる。

- 同じ時刻に実行する処理は同一のルールにまとめる。
- 共有された演算の処理結果は一時リレーションへ出力する。
- 共有された結合結果を最後に利用するルールでは一時リレーションをクリアする処理を行う。

これらを元に図 4 を ECA ルールに分割すると図 5 のように 4 つのルールが得られる。

R1 と R2 は Quote と News の配信単位が到着したときに Tag 演算を実行するルールである。特に R1 では、setTimer 演算によって R3, R4 のルールを起動させる時間にタイマーをセットしている。R3 は結合演算の処理と配信要求 1 に対する配信処理を行うルールである。R3 の発火する時刻は 18 時で、結合結果を一時リレーション (TempJoin) に格納し、共有する。また、配信要求 1 の配信スケジュールは 18 時であるので、このルール内で実行される。R4 は配信要求 2 のユーザに対して配信処理を行うルールで、22 時に実行される。このときすでに R3 によって統合結果が生成されているので、そこから配信要求 2 に必要な情報を取り出して配信する。また、このルールは一時リレーション TempJoin を最後に利用しているので、配信処理が終わった後、TempJoin の中身を削除する。

<b>Rule</b> R1	<pre> on: notify(NQuote) if: Tag(NQuote,     {(NQuote.name = 'A' ^ NQuote.price &gt; 400),      (NQuote.name = 'A' ^ NQuote.price &gt; 300)}) do: TempQuote += NQuote;     setTimer(next*.18:0:0(NQuote-ITS), timerR3);     setTimer(next*.22:0:0(NQuote-ITS), timerR4);                 </pre>
<b>Rule</b> R2	<pre> on: notify(NNews) if: Tag(NQuote,     {(NNews.name = 'A'), (NNews.name = 'A')}) do: TempNews += NNews;                 </pre>

<b>Rule</b> R3	<pre> on: notify(timerR3) if: true do: TempJoin += (     TempQuote     &lt;previous*.18:0:0(Quote.ITS)=previous*.18:0:0(News.ITS)     ^Quote.name=News.name     TempNews     &lt;Company.name=Quote.name     Company);     O1 = &lt;isNeeded(1,Quote.tag)^isNeeded(1,News.tag)     ^next*.18:0:0(Quote.ITS)=now(TempJoin);     Deliver(O1);                 </pre>
----------------	--

<b>Rule</b> R4	<pre> on: notify(timerR4) if: true do: O2 = &lt;isNeeded(2,Quote.tag)^isNeeded(2,News.tag)     ^next*.22:0:0(Quote.ITS)=now(TempJoin);     Deliver(O2);     TempJoin = {};                 </pre>
----------------	---

## 6 まとめと今後の課題

本稿では、配信型情報源統合環境において、複数の配信要求が与えられた際の処理方式について述べた。配信スケジュールなどの時間要件を考慮した上で演算の共有を行う点が本研究の特徴である。

今後の課題としては、配信時刻を考慮した結合演算の共有の判定手法の詳細化が挙げられる。

## 参考文献

- [1] 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.
- [2] M. Franklin and S. Zdonik. "Data in Your Face: Push Technology in Perspective," Proc. ACM SIGMOD, pp.516-519, Jun, 1998.
- [3] N. W. Paton and O. Diaz. "Active Database Systems," ACM Comp. Serv., 31(1), 1999.
- [4] 梶野智行, 北川博之, 石川佳治. "配信型情報源に対する統合要求記述と ECA ルール生成," 電子情報通信学会論文誌, Vol.J85-D-1, No.1, pp. 40-52, 2002 年 1 月.
- [5] 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.
- [6] P. Roy, S. Seshadri, S. Sudarshan, and S. Bhohe. "Efficient and Extensible Algorithms for Multi Query Optimization," ACM SIGMOD Conference, pp. 249-260, 2000.