D-029

XMLStream 向け検索言語からの VPA の生成
VPA Generation from Query Language for XML Stream

松田 達希† 内田 友樹† 藤田 悟‡
Tatsuki Matsuda Yuki Uchida Satoru Fujita

1. まえがき

近年,時々刻々と連続して発生するデータを即時に処理・分析を行う複合イベント処理(CEP)の研究が盛んに行われている. 連続して流れてくるデータはストリームデータと呼ばれ, その形式は様々である. そのため, 処理をより高速に, 効率的に行うためにはストリームデータの形式に合わせた CEP のエンジンを設計する必要がある. 我々は様々あるストリームデータの形式の中でも XML 形式に注目し, XML ストリームデータの高速な処理・解析を行う CEP エンジン, 及びエンジンを用いた高速解析システムの開発を行っている.

XML 形式のデータは構造化されており, データの検索や取得が容易に可能である. しかしながら, 従来型の問い合わせ言語は連続で発生し時間変化していく大量のストリームデータの観測・解析には適していない. そこで, Mozafari らは, 時系列上に発生するデータに対する検索要求を記述するための問い合わせ言語として, XSeq を開発した[1]. しかしながら, XSeq は一つのストリームしか扱うことができない.

本稿では複数の XML ストリームデータに対する問い合わせ言語 :Query Language for Multi-XML Stream(QLMXS)を設計する. また, QLMXS で記述された検索要求を解釈し, CEP エンジンのコアとして利用する Visibly Pushdown Automaton(VPA)[2]の生成を行う.

2. システムの概要

我々が提案するシステムの構成を図 1 に示す. 絶えず流れてくる XML ストリームデータは QLMXS エンジンで処理され, QLMXS エンジンは処理・解析結果をクライアントに渡す. QLMXS エンジンは検索エンジンのコアとして VPA を用いている. VPA はプッシュダウンオートマトンの制約を強めたものであり, XML や JSON などの入れ子構造のデータを容易にモデル化できるため, XML を効率的に処理できる[2]. 出力に関しては, XML の形式で出力させ, その結果を用いることで階層的または再帰的な解析を行うことも可能である.

3. 問い合わせ言語

3.1 QLMXS の仕様

我々が提案する問い合わせ言語:QLMXS は SQL や XPath, XSeq[1], Cayuga[2]を参考にしている. XML ストリームデータから単純なデータの検索・抽出だけでなく, 複数の XML に跨って解析を行うために複雑な条件を記述

† 法政大学大学院 情報科学研究科, Graduate School of Computer and Information Sciences, Hosei University
‡ 法政大学 情報科学部, Faculty of Computer and Information Sciences, Hosei University

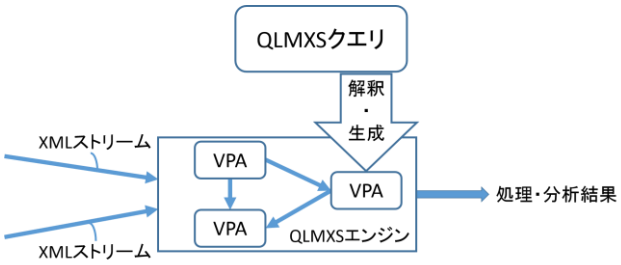

図 1 提案システムの構成

できるように設計を行った. QLMXS のクエリは基本的に, 出力する内容や形式を指定する return 節, 検索対象・条件や入力ストリームを指定する from 節, 変数を用いてデータのパス指定を行うための変数宣言を記述する with 節の 3 節から成り立つ. 詳しい記述方法に関しては, 具体的な例を挙げながら説明していく.

例として, 株式売買システムより各時刻における各企業の株情報が XML ストリームデータとして流れてくるような状況を想定する.

```
Example1. return stocks/stock/price >> stock_stream2
          from stocks/stock[price] << stock_stream
```

Example1 は, 流れてきた株情報から売値のみを抽出し, 新しいストリームとして出力するためのクエリである. 1 行目の return 節の中では, 前述した通り出力するデータや形式を指定している. return の後の stocks/stock/price は抽出するデータを XPath 形式で指定しており, その後の >> stock_stream2 では出力するストリームの名前を指定している. 2 行目の from 節では, from の後の stocks/stock[price]で検索対象・条件を XPath 形式で記述し, その後の<< stock_stream で入力ストリームを指定している. 即ち, Example1 は stock_stream の中から stocks タグの子タグである stock タグのさらに子タグに price タグが存在するならば, price タグ内のデータを抽出し, stock_stream2 として出力する, という意味になる. この例を, with 節を用いて変数を利用するように書き換えると Example2 のようになる.

```
Example2. return $X$Y/$Z >> stock_stream2
          from $X/$Y[$Z] << stock_stream
          with $X='stocks', $Y='stock', $Z='price'
```

Example2 と Example1 は同じ意味であるが, Example2 では変数を用いてパス指定をしている. 変数は接頭辞に$を付けて用いる. 複数宣言する場合には上記のようにカンマで区切って宣言する.

次に, 特定の条件を満たすデータの抽出を行う場合に関して, Example3 を用いて述べる.

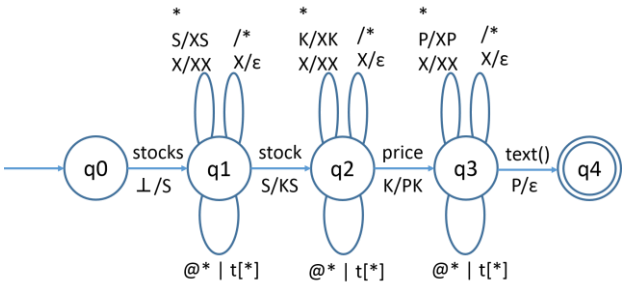

図 2　生成される VPA の例

Example3. return $X/$Y/$Z >> stock_stream2
from $X/$Y[$Z/text()>=100] << stock_stream
with $X='stocks', $Y='stock', $Z='price'

Example3 では, 株情報の中から売値が 100 以上であるものだけを抽出して出力するクエリである. from の後に $X/$Y[$Z/text()>=100]のように記述することで, タグ内の文字列に対する条件文を記述することができる. また, タグの属性に対する条件文や, 条件文の更なる追加は Example4 のように記述する.

Example4. return $X/$Y/$Z >> stock_stream2
from $X/$Y[$Z/text()>=100
and @name='A'] << stock_stream
with $X='stocks', $Y='stock', $Z='price'

Example4 は, Example3 の条件に加えて, 銘柄が A であるものの売値を抽出・出力するクエリである. 条件文を追加する場合は Example4 の 3 行目のように and あるいは or 等を用いて追加していく. また, タグの属性に関しては, 接頭辞に@を付けて指定する.

これらの基本的な検索・抽出に加え, いくつかの特徴的な記述方法に関しても説明する. QLMXS では XML ストリームを出力することが可能であるため, この出力そのものを入力として扱うことで階層的な記述が可能である. 例を Example5 に示す.

Example5. return $X/$Y >> stock_stream3
from $X[$Y/text()>=100] <<
(return $X/$Y
from $X/$Y[@name='A'] << stock_stream
with $X='stocks', $Y='stock')
with $X='stock', $Y='price'

Example5 は, 銘柄が A であるものを抽出したストリームから, 売値が 100 以上のものを抽出し, 出力するクエリである. 3 行目から 5 行目までが一つの QLMXS クエリであり, このクエリによる出力ストリームを入力ストリームとして 1,2,6 行目のクエリが処理している. 記述方法に関しては, 上記のように括弧の中に記述する. さらに, 複数の XML にわたって処理を行うための文法として, next 節を用いる. next 節を用いた例を Example6 に示す.

Example6. return $X/$Y#2 >> stock_stream3
from $X/$Y[@name='A'
and $Z/text()>=100] << stock_stream
next $X/$Y[@name='B'] << stock_stream2
with $X='stocks', $Y='stock', $Z='price'

Example6 は, 銘柄が A 且つ売値が 100 以上になった直後の, 銘柄が B である株の銘柄と売値を取得するための組み合わせクエリである. 4 行目の next 節において別のストリームデータに対する検索条件を指定している. これにより, 2 つの XML ストリームデータに跨った条件を記述できる. return 節におけるパス指定の後ろに#2 とあるが, これはどのストリームから抽出を行うのかを指定している. この例では, next 節で指定した 2 つ目のストリームを指す. このとき, next 節で指定するストリームは from 節で指定したストリームよりも時系列的に後のストリームである.

3.2 VPA への変換

QLMXS エンジンは QLMXS クエリから生成した VPA を用いて検索を実行する. VPA はスタックを持ち, 遷移時にスタックトップを参照しながら遷移先を決定する. よって, VPA を生成するためには, 状態の集合や遷移するためのシンボル集合, 遷移時にスタックに格納されるシンボルの集合, そして遷移関数が分かれば良い. 基本的にはクエリ中のパス指定の箇所を解釈し, どのタグが親子関係であるかを判別する等して VPA 生成に必要な情報を構築していく. Example1 から生成される VPA の例を図 2 に示す.

4. 考察

今回 QLMXS を設計するにあたって, Mozafari らによる XSeq[1]と, Demers らによる CEP システム:Cayuga[3]を参考にした. Cayuga は複数のストリームデータに跨った処理を可能としており, XSeq は XPath の拡張により XML ストリームデータに適した言語となっている. QLMXS では両者の特性を踏まえた設計を行った. また, クエリから VPA を生成することで, XML の高速な処理を可能としている. 生成した VPA に関しては今後最適化の研究を進める.

5. まとめ

本稿では, 我々が提案する CEP システムにおける問い合わせ言語:QLMXS の設計及び QLMXS クエリからの VPA の生成方法について述べた. QLMXS に関しては, 複数の XML ストリームにも対応した設計を行ったことで, 複雑な条件を比較的容易に記述することが可能となった. 今後の課題として, QLMXS の可読性の向上や生成した VPA の最適化などが挙げられる.

参考文献

[1] Mozafari B., Zeng K., Zaniolo C., "High-performance complex event processing over xml streams", Proceedings of the 2012 ACM SIGMOD International Conference on Management of Data, p.253-264 (2012).
[2] Alur R., Madhusudan P., "Visibly pushdown languages", Proceedings of the thirty-sixth annual ACM symposium on Theory of computing, p.202-211 (2004).
[3] Demers A., Gehrke J., Panda B., Riedewald M., Sharma V., White W., "Cayuga: A General Purpose Event Monitoring System", CIDR, Vol.7, p.412-422 (2007).