D-044

RDF ストリーム上での複合イベント検出
Complex Event Detection on RDF Streams

高橋 正和 † 築井 美咲 † 佐々木 勇和 ‡ 石川 佳治 †§
Masakazu Takahashi Misaki Yanai Yuya Sasaki Yoshiharu Ishikawa

1. はじめに

ICT の普及により，Web 上で生成されたデータやセンサによって取得された実世界のデータなど，様々な情報を取得できるようになってきている．特に，スマートフォンなどのモバイルデバイスと高速通信の普及で，GPS から取得した位置情報などのセンサデータを簡単に利用できるようになった．

このようにして取得したデータストリームからより高次のイベントを検出しようとする処理のことを**複合イベント処理**（Complex Event Processing, CEP）[1] と呼ぶ．既存の複合イベント処理システムの多くは，データを入力としてより高次のイベントを検出しているが，本研究ではイベントを入力としたシステムの開発を目指している．例えば，ある人の行動に関する情報を取得する場合，位置情報（緯度・経度）のみを入力とするのではなく，訪れた場所に関する情報やそのときの環境情報などを含むイベントを入力として扱えるようにする．入力をイベントとすることで，検出した複合イベントを再び入力イベントとして処理し，更に高次のイベントを検出することが可能となる．

イベントを表現するために，**Semantic Web** 技術 [2] を活用した**オントロジ**を利用する．オントロジを利用することで，複雑な知識体系の表現と意味的な処理が可能となる．また，公開されている既存のオントロジを再利用することもできる．オントロジは，データを主語・述語・目的語のトリプルで表現する枠組みである **RDF**（Resource Description Framework）で表現されているため，対象となるアプリケーションにおけるイベントも RDF データとして表現されることを想定する．

本研究では，RDF 形式のイベントストリームをオントロジを用いて処理し，ユーザが求める複合イベントを検出するシステムを提案する．本システムは，既存の RDF ストリーム処理エンジンを利用し，これに機能を追加することで実現する．以下では，FourSquare や Facebook などの**位置に基づくソーシャルネットワーク**（Location Based Social Networks, LBSN）[3] で発生するイベントの処理を例として，提案システムについて述べる．

2. イベント定義

本システムにおける「イベント」は，アプリケーションによって異なる意味をもつ．そこで，扱うイベントをすべて RDF で定義することで，様々なイベントを定義する．また，各イベント間の関係をオントロジで定義し，マッチングに利用する．

イベントの例として，ユーザが施設にチェックインする場合を考える．チェックインイベントを RDF 形式で表すと図 1 のようになる．`Checkin1` には，時間（`Instant1`），ユーザ名（`User1`），施設情報（`Feature1`）が含まれている．`Instant1` と `Feature1` の具体的な情報は，時間や地理空間に関するオントロジに従って RDF 形式で定義する．そして，チェックインイベントが発生するごとに図 1 のような RDF がシステムに送信される．

```
Checkin1   rdf:type              lbsn:checkin;
           lbsn:hasTime          Instant1;
           lbsn:hasAgent         User1;
           lbsn:hasFeature       Feature1.
Instant1   rdf:type              time:Instant;
           time:inXSDDateTime    "2014-06-30T15:00:00"
                                       ^^xsd:dateTime.
Feature1   rdf:type              lbsn:Bakery;
           lbsn:name             "Motoyama Bakery";
           geo:hasGeometry       Point1.
Point1     rdf:type              geo:Point;
           geo:asWKT             "POINT(35.163 136.962)"
                                       ^^sf:wktLiteral.
```

図 1: イベントの例

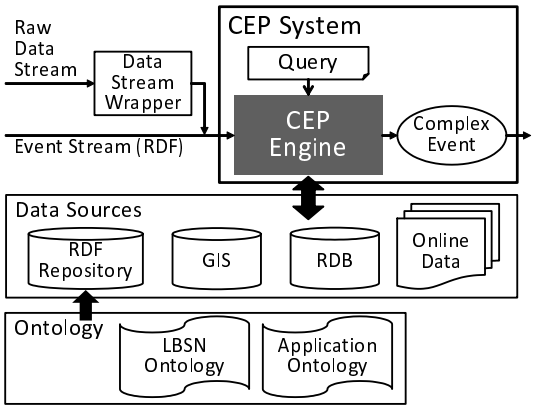

図 2: システムのアーキテクチャ

3. システムの概要

システムのアーキテクチャを図 2 に示す．

入力データは RDF 形式のイベントストリームを想定しているため，センサデータなどの RDF 形式以外のデータについては，**データストリームラッパ** で RDF ストリームに変換してシステムに入力する．例えば GPS から取得した位置情報のデータストリームを考えると，流れてくる位置情報を RDF 形式に変換するのに加え，時刻や GPS が搭載されている機器に関する情報など，アプリケーションに応じて必要な情報を付加して，イベントストリームを生成する．

入力 RDF ストリームはシステム内の **CEP エンジン**に送られ，問合せで指定されたイベントを検出する．CEP エンジンは必要に応じてさまざまな情報を外部から取得する．取得するデータには，オントロジで定義された知識が格納された RDF リポジトリや地理情報をもつ GIS，データベースなどの開発者が定義したローカルデータに加え，オンライン上に存在するデータを取得することを想定する．

4. CEP エンジン

CEP エンジンのアーキテクチャを図 3 に示す．

クエリマネジャは，システムから与えられた問合せを分析し，フィルタリングエンジンに登録する C-SPARQL[4] とイベント検出エンジンに登録する SPARQL[5] を生成する．SPARQL は，RDF に対するマッチングを行う SQL に類似の言語である．C-SPARQL は，RDF データストリーム上の連続的問合せのために SPARQL の拡張として提案された言語で，RDF ストリームに対して時間窓（Window）を指定し

†名古屋大学大学院情報科学研究科
Graduate School of Information Science, Nagoya University
‡名古屋大学未来社会創造機構
Institute of Innovation for Future Society, Nagoya University
§国立情報科学研究所
National Institute of Informatics

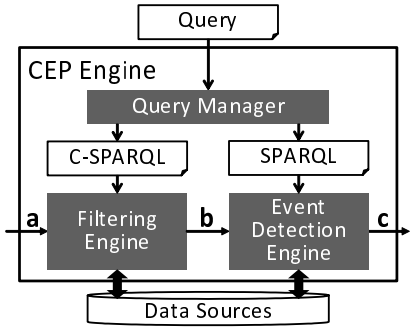

図 3: CEP エンジンのアーキテクチャ

```
 1:REGISTER QUERY NearFriend
 2:CONSTRUCT {
 3: ?checkin  rdf:type         lbsn:checkin;
 4:           lbsn:hasTime     ?time;
 5:           lbsn:hasAgent    ?person;
 6:           lbsn:hasFeature  ?feature.
 7:}
 8:FROM STREAM  <http://lbsn.org/checkin.trdf>
                          [RANGE 30m STEP 5m]
 9:WHERE {
10: ?checkin  rdf:type         lbsn:checkin;
11:           lbsn:hasTime     ?time;
12:           lbsn:hasAgent    ?person;
13:           lbsn:hasFeature  ?feature.
14: ?person   foaf:knows       USER.
15: ?feature    geo:hasGeometry ?geo.
16: BIND(geof:buffer(CURLOC, 200, uom:meter)
                                    as ?buff).
17: [a geo:Geometry ; geo:asWKT ?buff]
             geof:sfContains ?geo.
18:}
```

図 4: NearFriend 問合せ

て問合せを実行する機能や問合せ結果を RDF ストリームに整形して出力する機能などを有する．

フィルタリングエンジンは，RDF 形式のイベントストリーム（図 3-a）を入力とし，問合せで定義されたイベント検出に必要なイベント（図 3-b）を抽出する．フィルタリングエンジンとして，既存の RDF ストリーム処理エンジンである **C-SPARQL エンジン**を利用する．C-SPARQL エンジンで処理できることは限られている [6] ため，フィルタリングエンジンでは時間窓処理と RDF に対する簡単なマッチング処理のみを行う．

イベント検出エンジンは，フィルタリングエンジンから出力されたイベント（図 3-b）を入力とする．そして，地理空間演算（例：距離計算）などの処理を行い，結果を問合せで定義された形式に整形して出力する（図 3-c）．

5. CEP エンジンに対する問合せの例

5.1 NearFriend 問合せの概要

図 4 は，直近 30 分以内にユーザの近く（ここでは 200m とする）で発生した友人のチェックインイベントを検出し，検出結果を 5 分毎に RDF ストリームの形式で出力する NearFriend 問合せである．

`FROM STREAM` 句では，チェックインイベントのストリームを取得し，`[RANGE 30m STEP 5m]` で大きさ 30 分で 5 分毎にスライドする時間窓を指定している．

`WHERE` 句では，各チェックインイベントが問合せの条件に当てはまるか判断する．友人であるかの判断は，FOAF（Friend of a Friend）[7] の `foaf:knows` という述語で行っている．近くにいるかどうかは，ユーザの現在地から半径 200m の範囲を表す `?buff` を作成し，その範囲に友人が存在しているかを `geof:sfContains` という GeoSPARQL[8] の述語を使用して判断している．

結果は，`CONSTRUCT` 句で指定された RDF ストリーム形式に整形される．

5.2 問合せ処理

はじめに，NearFriend 問合せを与えられたクエリマネジャは，問合せを分析して C-SPARQL と SPARQL を生成する．C-SPARQL は `FROM STREAM` 句（図 4 の 8 行目）と `WHERE` 句の一部（9〜15 行目）に該当する問合せで，フィルタリングエンジンに登録される．一方，SPARQL は `WHERE` 句の残り（16〜18 行目）と `CONSTRUCT` 句（2〜7 行目）に該当する問合せで，イベント検出エンジンに登録される．

入力イベントストリーム（図 3-a）は，図 1 のようなチェックインイベントのストリームとする．フィルタリングエンジンは，入力に対し C-SPARQL で定義された問合せを実行し，「30 分以内に発生した友人のチェックインイベント」を入力イベントと同じ形式で出力する（図 3-b）．

イベント検出エンジンは，フィルタリングエンジンから出力されたイベントに対し，各チェックインイベントがユーザの近くで発生したものかを判断する SPARQL 問合せを実行する．そして，問合せ結果を `CONSTRUCT` 句で指定されたチェックインイベントストリームの形式に整形して出力する（図 3-c）．

6. まとめと今後の課題

本稿では，オントロジを利用したイベント処理システムの概要について述べた．本システムは，オントロジを利用してイベントを動的に定義し，関連イベントも含めてマッチングを行うことができる．オントロジに基づくイベント処理に関する議論は [9] でなされているが，動画像データ等を対象とした知識表現が研究の対象であり，本研究とは目的が異なる．

C-SPARQL をはじめとする RDF ストリーム処理エンジンの研究は現在も続けられているため，仕様の変更や機能の追加が行われる可能性がある．今後はこのような研究の動向を注視しつつ，システムの詳細化と実装について取り組む予定である．

謝辞

本研究は科研費（25280039, 26540043）による．

参考文献

[1] Gianpaolo Cugola and Alessandro Margara. Processing Flows of Information: From Data Stream to Complex Event Processing. *ACM Computing Surveys*, Vol. 44, No. 3, 2012.

[2] 荒木雅弘. フリーソフトで学ぶセマンティック Web とインタラクション. 森北出版, 2010.

[3] Chi-Yin Chow, Jie Bao, and Mohamed F. Mokbel. Towards Location-based Social Networking Services. In *2nd Intl. Workshop on Location Based Social Networks (LBSN'10)*, pp. 31–38, 2010.

[4] Davide Francesco Barbieri, Daniele Braga, Stefano Ceri, Emanuele Della Valle, and Michael Grossniklaus. Querying RDF Streams with C-SPARQL. *SIGMOD Rec.*, Vol. 39, No. 1, pp. 20–26, September 2010.

[5] SPARQL 1.1 overview. http://www.w3.org/TR/sparql11-overview/, March 2013.

[6] Ying Zhang, Pham Minh Duc, Oscar Corcho, and Jean-Paul Calbimonte. SRBench: A Streaming RDF/SPARQL Benchmark. In *Proceedings of the 11th International Conference on The Semantic Web - Volume Part I*, ISWC'12, pp. 641–657, Berlin, Heidelberg, 2012. Springer-Verlag.

[7] Dan Brickley and Libby Miller. FOAF vocabulary specification 0.98, August 2010.

[8] Robert Battle and Dave Kolas. Enabling the geospatial semantic web with Parliament and GeoSPARQL. *Semantic Web*, Vol. 3, No. 4, pp. 355–370, 2012.

[9] A. Gupta and R. Jain. Managing Event Information: Modeling, Retrieval, and Applications. *Morgan & Claypool*, 2011.